Analytical Model of Coding-Based Reprogramming Protocols in Lossy Wireless Sensor Networks

Jun-Wei Li*, Shi-Ning Li*, Yu Zhang*, Yee Wei Law†, Xingshe Zhou* and Marimuthu Palaniswami†
*School of Computer Science and Technology, Northwestern Polytechnical University, China
†Department of Electrical and Electronic Engineering, The University of Melbourne, Australia
Email: ljw@mail.nwpu.edu.cn, {lishining, zhangyu, zhouxs}@nwpu.edu.cn, {ywlaw, palani}@unimelb.edu.au

Abstract—**Multi-hop over-the-air reprogramming is essential for the remote installation of software patches and upgrades in wireless sensor networks (WSNs). Recently, coding-based reprogramming protocols are proposed to address efficient code dissemination in environments with high packet loss rate. The problem of analyzing the performance of these protocols, however, has not been explored in the literature. In this paper, we present a high-fidelity analytical model based on Dijkstra's shortest path algorithm to measure the completion time of coding-based reprogramming protocols. Our model takes into account not only page pipelining and negotiation, but also coding computation. Results from extensive simulations of a representative coding-based reprogramming protocol called Rateless Deluge are in good agreement with the performance predicted by our model, thus validating our approach. Our analytical results show both the number of packets per page and the finite field size have significant impact on completion time. Most notably, the time overhead of coding computation exceeds that of communication when the number of packets per page is 24 and the finite field size is at least 2^4.**

I. Introduction

As application requirements evolve, the software on sensor nodes needs to be updated, or patched for errors. In cases where physical access to these nodes is limited post-deployment, or manually updating these sensor nodes is prohibitively costly, multi-hop over-the-air reprogramming presents an ideal solution. Conventional reprogramming protocols–with Deluge being the current benchmark [1]–suffer from steep performance degradation when packet loss rate is high, as a result of interference, jamming, or natural environmental factors. Recently, rateless coding schemes have been introduced to improve the *resilience* of reprogramming protocols to packet loss [2]–[7], i.e., when packet loss is substantial, coding-based reprogramming protocols require significantly less time and energy to complete than traditional reprogramming protocols. However, there is currently no analytical model for quantifying the performance of coding-based reprogramming protocols. An analytical model is important because (i) it provides an economical alternative to labor-intensive simulations and field experimentations for performance quantification; (ii) it sheds valuable insights into the "mechanics" of the protocols, thereby providing essential clues for ongoing improvements of the protocols.

To the best of our knowledge, we are the first to propose an analytical model based on Dijkstra's shortest path algorithm for coding-based reprogramming protocols. For validation, we apply our model to Rateless Deluge [2], which is an extension of Deluge using random linear codes. Results from extensive simulations of Rateless Deluge are in good agreement with the performance predicted by our model, thus validating our approach. Our main contributions are as follows.

(a) Our model is applicable to any network topology.
(b) Our comprehensive model is based on a thorough analysis of three pivotal features of a coding-based programming protocol: meta-data negotiation, page pipelining, and coding computation.
(c) Using our validated model, we demonstrate that it is essential to take a joint consideration of the finite field size and the number of packets per page to lessen the completion time.

The remainder of this paper is organized as follows. Section II reviews related work on performance analysis of reprogramming protocols for WSNs. In Section III, we provide the problem statement and an overview of our approach. Section IV presents the core components of our model: network propagation and per-hop operations. In Section V, we discuss the coding part of per-hop operations in the context of Rateless Deluge. Details on experiments and evaluation results are provided in Section VI. Section VII concludes the paper.

II. Related work

Mathematical analysis of the performance of WSN protocols is challenging, but research interest is growing in this area [8]. In the area of reprogramming protocols, some preliminary work on performance modeling has been conducted. Hui et al. [1] is the first to analyze the performance of Deluge, a *non*-coding-based reprogramming protocol. Moreover, their model is only applicable to linear topologies. In comparison, by the virtue of Dijkstra's shortest path algorithm, our model is applicable to any network topology.

The analysis in [9] and [10] ignores either page pipelining or other protocol details. Dong et al.'s work [11] is the closest to ours. Their analytical model is applicable to any topology, and they have established a relationship between packet reception ratio (PRR) and number of request packets. We propose refinement to their result on the relationship between PRR and number of request packets. The most significant difference between existing work and ours is that we present a thorough analysis of coding operation in a concrete coding-based reprogramming protocol, and establish an analytical model to evaluate it in terms of completion time.

Starobinski et al. [12] propose a *theoretical framework* to evaluate the asymptotic performance limits of data dissemination in multi-channel, single-radio WSNs. In comparison, the target of our model is concrete coding-based reprogramming protocols for the more common single-channel, single-radio WSNs.

III. Problem statement and approach outline

The problem of determining the completion time of code dissemination in the whole network can be formalized as such. Among m sensor nodes $\{S_1, S_2, \ldots, S_m\}$ in a network, S_1 is designated as the base station that disseminates a code image to all other sensor nodes. A code image is divided into n pages $\{C_1, C_2, \ldots, C_n\}$. Let us denote the completion time for sending page C_k from node S_i to node S_j by $T(k, i, j)$, where $k \in [1, n]$ and $i, j \in [1, m]$. Then, the completion time of code dissemination is $\max_j T(n, 1, j)$. In general, $T(k, i, j)$ depends on:

(1) the propagation path from S_i to S_j;
(2) the duration of per-hop operations for all sensor nodes along the propagation path.

Therefore, the completion time of C_i on each sensor node along the propagation path has to be determined. Furthermore, the time cost of all protocol behaviors, such as coding computation, page pipelining has to be determined.

Our analytical model consists of two main components: *network propagation* and *per-hop operations*. The "network propagation" component captures the dynamics of a code image being disseminated across a network, and takes input from the "per-hop operations" component. The "per-hop operations" component captures the complexity of meta-data negotiation and coding computation, and takes input from the physical layer model, which encapsulates physical parameters such as packet reception ratio (PRR), receiver signal strength (RSSI), and link quality (LQI). These physical parameters can be obtained from simulations, as shown in Section VI-A.

IV. The analytical model

In this section, we first present the network propagation part, and then the per-hop operations part of our model.

Network propagation: The completion time of any page C_k from S_1 to S_j, denoted $T(k, 1, j)$, depends on the propagation path from S_1 to S_j. We employ Dijkstra's shortest path algorithm [13] to determine the propagation path as noted in [12], [14]. Suppose the path l from S_1 to S_j consists of L nodes, and we label the nodes as $S_{l(1)}, S_{l(2)}, \ldots, S_{l(p)}, \ldots, S_{l(L)}, l(p) \in [1, m], p \in [1, L]$. To determine $T(k, 1, l(p))$, we first observe

$$T(k, 1, l(p)) \geq T(k, 1, l(p-1)) + T(k, l(p-1), l(p)), \quad (1)$$

where $T(i, l(p-1), l(p))$ represents the duration of per-hop operations, including the sending of C_k from $S_{l(p-1)}$ to $S_{l(p)}$, $k \geq 1, p \geq 2$. The equality in (1) is due to the fact that page C_k is propagated from node $S_{l(p-1)}$ to node $S_{l(p)}$, whereas the inequality in (1) is due to the fact that not all packets in page C_k arrive through the shortest path. Next, we consider page pipelining (also called spatial multiplexing [1]), which is the concurrent transmission of different pages by different nodes. The nodes must be at least *three hops* apart to avoid collisions. In the ideal scenario of page pipelining, when the node $S_{l(p+3)}$ finishes receiving page C_{k-1}, the node $S_{l(p)}$ also finishes receiving page C_k. In other words, for $k \geq 2$,

$$T(k, 1, l(p)) \geq T(k-1, 1, l(q)), \ q = \min(p+3, L). \quad (2)$$

Approximating $T(k, 1, l(p))$ by its lower bound, we combine (1) and (2) to obtain, for $p \geq 2$,

$$\begin{aligned} T(k, 1, l(p)) = \max \big\{ & T(k, 1, l(p-1)) + T(k, l(p-1), l(p)), \\ & T(k-1, 1, l(\min(p+3, L))) \big\}. \end{aligned} \quad (3)$$

The maximum of all shortest paths, $\max_j T(n, 1, j)$, is calculated recursively according to (3), with the initial condition $T(k, 1, l(1)) = 0$.

Per-hop operations: We consider the expected duration of per-hop operations in terms of contribution from communication and computation, i.e., $E(T(k, l(p-1), l(p))) = E(T_{\text{comm}}) + E(T_{\text{comp}})$. The communication overhead is due to meta-data negotiation. The meta-data negotiation mechanism established by SPIN [15], and adopted by Deluge and its derivatives (e.g., Rateless Deluge), enables the epidemic-like propagation of a code image from the base station to all sensor nodes. A firmware is disseminated page by page. A sensor node possessing a new page broadcasts an advertisement (ADV) that includes the version number of the new code image, the number of pages available for that image, and the number of the new page. Receivers of the advertisement compare the advertised page number with their existing page number, and send requests (REQ) for the new page if their page number is lower. Advertisements are suppressed within a one-hop range to avoid redundant transmissions. When a node exceeds its limit of λ request packets, it must wait for another advertisement before making additional requests. The advertiser starts broadcasting data packets (DATA) to the requesters upon request. Additionally, data packets are encoded before transmission in coding-based reprogramming protocols. Therefore from a receiver's perspective, $E(T_{\text{comm}})$ includes the overhead for the reception of advertisement packets, the transmission of request packets, and the reception of data packets. In other words,

$$E(T_{\text{comm}}) = E(T_{\text{adv}}) + E(T_{\text{req}}) + E(T_{\text{data}}). \quad (4)$$

The expected time of advertisement reception can be estimated as

$$E(T_{\text{adv}}) = E(N_{\text{pkt}}) \cdot E(r_i) \cdot (1 + E(N_{\text{supp}})), \quad (5)$$

where $E(N_{\text{pkt}}) = \frac{1}{\text{PRR}}$ is the expected number of transmissions for one packet, at a packet reception ratio of PRR. Measurements of PRR and its correlation properties will be discussed in Section VI-A. $E(r_i)$ specifies the expected wait time between advertisement messages. $E(N_{\text{supp}})$ specifies the number of times that an advertisement is suppressed by the previous node on the propagation path before transmission. We set $E(N_{\text{supp}}) = 1$ as in the Deluge protocol.

The expected time of request transmission can be estimated as

$$E(T_{\text{req}}) = E(N_{\text{pkt}}) \cdot E(N_{\text{reps}}) \cdot \frac{\tau}{2} + E(T_{\text{back}}), \qquad (6)$$

where $E(N_{\text{pkt}})$ is the same as in (5). $E(N_{\text{reps}})$ specifies the expected number of request packets needed to receive one page. Existing work [1], [14] either fixes $E(N_{\text{reps}})$ as a constant, or experimentally finds its relationship with PRR. However, our empirical observation in Section VI-A shows that $E(N_{\text{reps}})$ is correlated with PRR. $\frac{\tau}{2}$ specifies the expected duration of random backoff before request transmission. $E(T_{\text{back}})$ specifies the expected time for additional advertisements and is defined as

$$E(T_{\text{back}}) = E(N_{\text{pkt}}) \cdot \left(\frac{E(N_{\text{reps}})}{\lambda} - 1 \right) \cdot E(T_{\text{adv}}), \qquad (7)$$

where λ is the number of requests, more than which the node has to wait for another advertisement before making additional requests.

The expected time of data reception can be estimated as

$$E(T_{\text{data}}) = E(N_{\text{pkt}}) \cdot E(T_{\text{pkt}}) \cdot N_{\text{ppp}} \cdot E(N_{\text{mac}}), \qquad (8)$$

where $E(N_{\text{pkt}})$ is the same as in (5). $E(T_{\text{pkt}})$ specifies the transmission time for one packet. N_{ppp} is the number of packets per page. $E(N_{\text{mac}})$ captures the expected time of MAC-layer delay due to contention.

For computation overhead, we consider only the most time-consuming operations: encoding and decoding, i.e., $E(T_{\text{comp}}) = E(T_{\text{enc}}) + E(T_{\text{dec}})$. As different coding-based reprogramming protocols use different coding schemes, we quantify $E(T_{\text{enc}})$ and $E(T_{\text{dec}})$ for Rateless Deluge only in this work, the detail of which is given in the next section.

V. Computation overhead of Rateless Deluge

In Rateless Deluge, random linear codes are used for encoding/decoding data packets. To send a page consisting of packets $X_1, \ldots, X_{N_{\text{ppp}}}$, the packets are encoded as $Y_1, \ldots, Y_{N_{\text{enc}}}, N_{\text{enc}} = 1.5 N_{\text{ppp}}$ according to (9):

$$\begin{bmatrix} \beta_{1,1} & \cdots & \beta_{1,N_{\text{ppp}}} \\ \beta_{2,1} & \cdots & \beta_{2,N_{\text{ppp}}} \\ \vdots & \ddots & \vdots \\ \beta_{N_{\text{enc}},1} & \cdots & \beta_{N_{\text{enc}},N_{\text{ppp}}} \end{bmatrix} \begin{bmatrix} X_{1,j} \\ X_{2,j} \\ \vdots \\ X_{N_{\text{ppp}},j} \end{bmatrix} = \begin{bmatrix} Y_{1,j} \\ Y_{2,j} \\ \vdots \\ Y_{N_{\text{enc}},j} \end{bmatrix}, \qquad (9)$$

where $X_{i,j}$ denotes the jth byte of X_i, and β's are pseudorandom numbers. By populating the matrix on the left hand side with pseudorandom numbers, the matrix is rendered almost surely non-singular. Decoding $Y_1, \ldots, Y_{N_{\text{enc}}}$ to $X_1, \ldots, X_{N_{\text{ppp}}}$ is by means of Gaussian elimination.

A page can be pre-coded in anticipation of future requests to reduce dissemination delay. The expected time of encoding, taking pre-coding into account, is thus

$$E(T_{\text{enc}}) = E(\mu_{\text{pre}}) \cdot E(N_{\text{enc}}) \cdot E(T_{\text{enc.pkt}}), \qquad (10)$$

where $E(\mu_{\text{pre}})$ specifies the ratio that time overhead could be reduced due to pre-coding, which is constrained by RAM size, page size N, and delay before pre-coding. $E(N_{\text{enc}}) = 1.5\, N_{\text{ppp}}$ specifies the expected number of packets to be encoded, which is proportional to N_{ppp} in Rateless Deluge. N_{ppp} is the number of packets per page. A packet is encoded byte by byte, and each byte needs $(5q\,\text{AND} + 2q\,\text{XOR} + 4q\,\text{CMP} + 4q\,\text{SHIFT})$ operations, which is estimated from public-accessible source code of Rateless Deluge protocol [2]. The expected time to encode one packet is thus

$$E(T_{\text{enc.pkt}}) = \frac{N_{\text{ppp}}}{\lfloor \frac{q+7}{8} \rfloor} \cdot (5q \cdot \tau_{\text{AND}} + 2q \cdot \tau_{\text{XOR}} + 4q \cdot \tau_{\text{SHIFT}} + 4q \cdot \tau_{\text{CMP}}), \qquad (11)$$

where the finite field size is by default $2^q = 2^8$ in Rateless Deluge. τ_{AND}, τ_{XOR}, τ_{SHIFT}, and τ_{CMP} are byte-wise arithmetic and logical operations.

The decoding procedure in Rateless Deluge consists solely of Gaussian elimination. The expected time of decoding is thus

$$E(T_{\text{dec}}) = E(N_{\text{dec}}) \cdot E(T_{\text{gauss}}), \qquad (12)$$

where $E(N_{\text{dec}}) = \frac{1}{1 - Pr(\text{dec.fail})}$ specifies the expected number of times of decoding, under a decoding failing probability of $Pr(\text{dec.fail})$, which depends on N_{ppp} and the finite field size 2^q. $E(T_{\text{gauss}})$ specifies the expected duration of Gaussian elimination and is estimated as

$$E(T_{\text{gauss}}) = \frac{N_{\text{ppp}}^2 + 3N_{\text{ppp}} - 1}{3} \cdot T_{\text{enc.pkt}} + \frac{4N(2N_{\text{ppp}}^2 + 3N_{\text{ppp}} - 5)}{3} \cdot \tau_{\text{SUB}}, \qquad (13)$$

where $T_{\text{enc.pkt}}$ is as defined in (11), and τ_{SUB} is a byte-wise arithmetic operation.

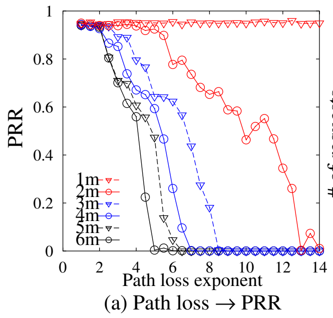
VI. Experiments and evaluations

Usage of our analytical model necessitates the empirical determination of a few parameters. The purpose of simulations is hence twofold: (i) to determine or "set up" these parameters; (ii) to provide a benchmark for the accuracy and efficiency assessment of the analytical model. We use the discrete event simulator TOSSIM to simulate Rateless Deluge and a secure version of Rateless Deluge called Sreluge [16]. As part of our results, we make a new observation on the relationship between the finite field size 2^q and the number of packets per page N_{ppp} of Rateless Deluge.

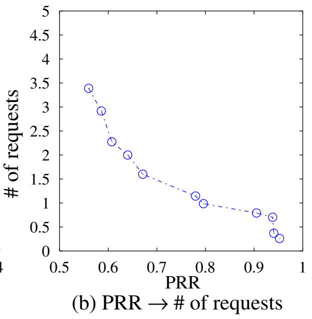
A. Parameters Setup

We set the maximum random backoff time, $\tau = 0.5$ seconds, and set $\lambda = 2$ such that a node makes a maximum of 2 requests after an advertisement (see (7)). Through simulations, we are able to determine the following parameter values:

- average wait time between advertisement messages, $E(r_i) = 0.8$ seconds;
- average transmission time for one packet, $E(T_{\text{pkt}}) = 0.00862$ seconds.



(a) Path loss $\rightarrow$ PRR



(b) PRR $\rightarrow$ # of requests

Fig. 1. (a) Plot of PRR against path loss exponent. (b) Plot of number of requests against PRR.

The value of $E(N_{\text{reps}})$ depends on the PRR. To obtain PRR, we measure RSSI and LQI [17]. In TOSSIM, the log-normal shadowing channel model [18] is used to model radio connectivity between nodes. Given a reference distance D_0 (in meters), and the corresponding path loss PL_{D_0} (in dB), the link quality between nodes can be calculated, given the path loss exponent, shadowing deviation, and noise floor (in dBm). Experiments are conducted by running the *CountRadio* application from TinyOS on two nodes that are separated from each other with variable node spacing and path loss exponent. We program each node to send 1,000 packets to and receive from the other node, and count the number of received packets. The PRR is estimated as

$$\text{PRR} = \frac{\sum_{i=1}^{N_{\text{exp}}} p_i - \max_{i=1}^{N_{\text{exp}}} p_i - \min_{i=1}^{N_{\text{exp}}} p_i}{(N_{\text{exp}} - 2)\sigma}, \tag{14}$$

where N_{exp} specifies the number of experiments conducted, $\sigma = 1000$ describes the number of packets sent, and p_i specifies the number of packets received successfully in the ith experiment. Here we exclude minimum and maximum values due to random noise. The results in Fig. 1 are consistent with intuition:

- PRR drops as path loss exponent increases (see Fig. 1(a)). We note that when node spacing approaches reference distance D_0 (= 1 m), the results are unusable due to the limitation of the radio model.
- Average number of request packets $E(N_{\text{reps}})$ increases as PRR decreases. In Fig. 1(b), the PRR values are taken from Table I. When PRR is high (over 80%), most nodes send no requests as data packets are overheard in advance. $E(N_{\text{reps}})$ drops slowly when PRR is small (less than 80%), but drops sharply when PRR is over 80%.

Hence, given a node spacing value and path loss exponent, we would be able to determine PRR and $E(N_{\text{reps}})$.

B. Model Validation and Evaluation

To validate our analytical model, we use the firmware of the *Blink* application from TinyOS, consisting of 16 pages with 168 bytes in each page, for reprogramming.

Accuracy: We use TOSSIM to simulate a 10×10 grid network, with a path loss exponent of 4.0. Node spacing is varied from 1.5m to 6.0m at an increment of 0.5m. Figure 2 compares the proposed model with simulation results. The analytical and simulation results are in good agreement for both the cases when radio connectivity is poor (large node spacing) and when radio connectivity is excellent (small node spacing). Furthermore, we make extensive comparisons with Sreluge in the absence of attacks and the results are similar to Fig. 2.

Efficiency: We implement the proposed model on HP EliteBook 8440p laptop in C. The laptop has a quad 2.27 GHz CPU with 8 GB RAM and runs Linux. We conduct this experiment with a path loss exponent of 4.0 and node spacing of 5.0 m. Table II compares the time cost of analytical calculations and simulations corresponding to different network sizes. For a 30×30 grid network, the analytical model finishes in less than 2 seconds, while the simulation costs more than 1 hour.

C. Observation on q and N_{ppp}

Finally, we explore how the size 2^q of the finite field $GF(2^q)$, and the number of packets per page N_{ppp} affect performance of coding-based reprogramming protocols. We use the code image of the *Blink* application for reprogramming.

TABLE I
PRR MEASUREMENTS IN A 10X10 GRID NETWORK

node spacing	PRR	node spacing	PRR
1.5m	0.9405	2.0m	0.9380
2.5m	0.9050	3.0m	0.7959
3.5m	0.7796	4.0m	0.6711
4.5m	0.6402	5.0m	0.6075
5.5m	0.5858	6.0m	0.5597

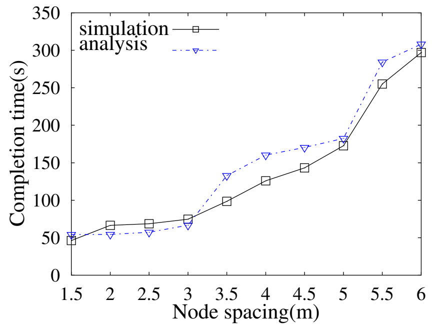


Fig. 2. Plots of completion time against node spacing: analytical model vs simulations.

TABLE II
COMPARISON OF TIME COST TO CALCULATE COMPLETION TIME

network size	simulation(s)	analytical model(s)
5 × 5	26.308	0.015
10 × 10	110.425	0.023
15 × 15	204.493	0.130
20 × 20	791.483	0.340
25 × 25	2308.209	0.755
30 × 30	5897.203	1.526

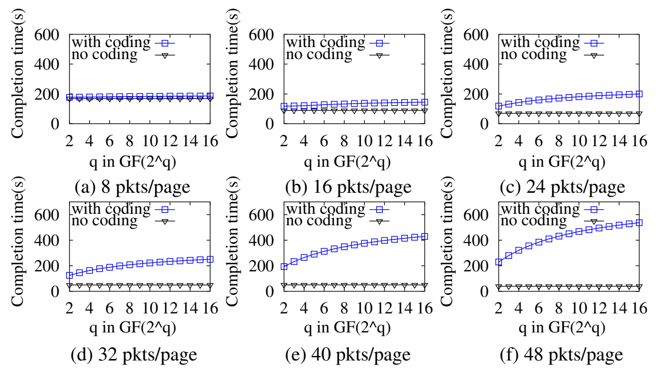


Fig. 3. Analytical results showing the impact of the finite field size and number of packets per page.

We choose $q \in [2, 16]$, and $N_{\text{ppp}} \in \{8, 16, 24, 32, 40, 48\}$. We note that when q equals 16, decoding needs 65KB memory space to store inverses; this requirement exceeds the RAM size of most sensor node platforms (including TelosB which has 10 KB RAM) [2]. We calculate the completion time for both cases when coding computation is accounted for and when it is not, and the results are shown in Fig. 3.

Case where coding computation is accounted for: When N_{ppp} is fixed, the completion time increases with q, because encoding complexity is positively correlated with q. When q is fixed, the completion time on overall increases with N_{ppp} but experiences a temporary decline for $8 \leq N_{\text{ppp}} \leq 16$. This discovery is contrary to the incorrect intuition that the completion time increases continuously with N_{ppp} (more packets to encode/decode) for a fixed q.

Case where coding computation unaccounted for: For a fixed q, the completion time declines as N_{ppp} increases. This is because as the number of requests is reduced, so is the time wasted in waiting for and receiving advertisements.

Furthermore, when $N_{\text{ppp}} = 24$ and $q \geq 4$, the time overhead of coding computation exceeds that of communication. Additionally, when $N_{\text{ppp}} = 16$, either a decrease or an increase in N_{ppp} will cause a significant increase in the completion time. When $N_{\text{ppp}} \geq 32$, a decrease in q is accompanied with a significant decline in completion time. Based on the above observation, the combination $N_{\text{ppp}} = 16$ and $q = 8$ provides close-to-minimum completion time and ease of implementation on existing WSN architectures.

VII. Conclusion

An analytical model serves as an economical alternative to computationally expensive simulations and labor-intensive field experimentations for the performance evaluation of coding-based reprogramming protocols. Moreover, it provides essential clues for ongoing improvements of the protocols. To measure the completion time of coding-based reprogramming protocols, we propose a high-fidelity analytical model that is applicable to any network topology. Our model takes into account not only page pipelining and negotiation, but also coding computation. Results from extensive simulations of a representative coding-based protocol called Rateless Deluge confirms the validity of our model. Our analytical results show that the time overhead of coding computation exceeds that of communication when the number of packets per page N_{ppp} is 24 and the finite field size 2^q is at least 2^4.

Acknowledgement

This work is supported by National Science and Technology Major Project of China under grant No. 2012ZX03005007. Yu Zhang is supported by NPU Foundation for Fundamental Research under grant No. JC20110268. Yee Wei Law is supported by ARC DP1095452 and EU FP7-ICT-2009-5-257992.

References

[1] J. W. Hui and D. Culler, "The dynamic behavior of a data dissemination protocol for network programming at scale," in *Proc. ACM SenSys'04*, Baltimore, MD, USA, Nov. 2004, pp. 81–94.

[2] A. Hagedorn, D. Starobinski, and A. Trachtenberg, "Rateless deluge: Over-the-air programming of wireless sensor networks using random linear codes," in *Proc. ACM/IEEE IPSN'08*, Apr. 2008, pp. 457–466.

[3] I.-H. Hou, Y.-E. Tsai, T. F. Abdelzaher, and I. Gupta, "Adapcode: Adaptive network coding for code updates in wireless sensor networks," in *Proc. IEEE INFOCOM'08*, Apr. 2008, pp. 1517–1525.

[4] M. Rossi, G. Zanca, L. Stabellini, R. Crepaldi, A. F. Harris, and M. Zorzi, "Synapse: A network reprogramming protocol for wireless sensor networks using fountain codes," in *Proc. IEEE SECON'08*, Jun. 2008, pp. 188–196.

[5] A. D. Wood and J. A. Stankovic, "Online coding for reliable data transfer in lossy wireless sensor networks," in *Proc. IEEE DCOSS'09*, Jun. 2009.

[6] M. Rossi, N. Bui, G. Zanca, L. Stabellini, R. Crepaldi, and M. Zorzi, "Synapse++: Code dissemination in wireless sensor networks using fountain codes," *IEEE Trans. Mobile Comput.*, vol. 9, no. 12, pp. 1749–1765, Dec. 2010.

[7] W. Dong, C. Chen, X. Liu, J. Bu, and Y. Gao, "A lightweight and density-aware reprogramming protocol for wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 10, no. 10, pp. 1403–1415, Oct. 2011.

[8] Y. Wang, M. C. Vuran, and S. Goddard, "Cross-layer analysis of the end-to-end delay distribution in wireless sensor networks," *IEEE/ACM Trans. Netw.*, vol. 20, no. 1, pp. 305–318, Feb. 2012.

[9] R. K. Panta, I. Khalil, and S. Bagchi, "Stream: Low overhead wireless reprogramming for sensor networks," in *Proc. IEEE INFOCOM'07*, Anchorage, AK, USA, May 2007, pp. 928–936.

[10] P. De, Y. Liu, and S. K. Das, "An epidemic theoretic framework for vulnerability analysis of broadcast protocols in wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 8, no. 3, pp. 413–425, Mar. 2009.

[11] W. Dong, C. Chen, X. Liu, G. Teng, J. Bu, and Y. Liu, "Bulk data dissemination in wireless sensor networks: Modeling and analysis," *Computer Networks*, vol. 56, no. 11, pp. 2664–2676, Jul. 2012.

[12] D. Starobinski and W. Xiao, "Asymptotically optimal data dissemination in multichannel wireless sensor networks: Single radios suffice," *IEEE/ACM Trans. Netw.*, vol. 18, no. 3, pp. 695–707, Jun. 2010.

[13] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. Cambridge, MA: The MIT Press, 2001.

[14] W. Dong, C. Chen, X. Liu, J. Bu, and Y. Liu, "Performance of bulk data dissemination in wireless sensor networks," in *Proc. IEEE DCOSS'09*, Marina del Rey, CA, USA, Jun. 2009.

[15] W. R. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive protocols for information dissemination in wireless sensor networks," in *Proc. ACM/IEEE MobiCom'99*, 1999, pp. 174–185.

[16] Y. W. Law, Y. Zhang, J. Jin, M. Palaniswami, and P. Havinga, "Secure Rateless Deluge: Pollution-Resistant Reprogramming and Data Dissemination for Wireless Sensor Networks," *EURASIP J. Wirel. Commun. Netw.*, vol. 2011, Jan. 2011.

[17] K. Srinivasan, P. Dutta, A. Tavakoli, and P. Levis, "Understanding the causes of packet delivery success and failure in dense wireless sensor networks," in *Proc. ACM SenSys'06*, 2006, pp. 419–420.

[18] M. Zuniga and B. Krishnamachari, "Analyzing the transitional region in low power wireless links," in *Proc. IEEE SECON'04*, Santa Clara, CA, USA, Oct. 2004, pp. 517–526.