IN-NETWORK PROCESSING FOR MISSION-CRITICAL WIRELESS NETWORKED SENSING AND CONTROL: A REAL-TIME, EFFICIENCY, AND RESILIENCY PERSPECTIVE

by

QIAO XIANG

DISSERTATION

Submitted to the Graduate School

of Wayne State University,

Detroit, Michigan

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

2014

MAJOR: COMPUTER SCIENCE

Approved by:

Advisor Date

DEDICATION

To my family and friends

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

After the past decade of active research and field trials, wireless sensor networks (which we call *sensornet*s interchangeably) have started penetrating into many areas of science, engineering, and our daily life. They are also envisioned to be an integral part of cyber-physical systems such as those for alternative energy, transportation, and healthcare. In supporting mission-critical, real-time, closed loop sensing and control, CPS sensornets represent a significant departure from traditional sensornets which usually focus on open-loop sensing, and it is critical to ensure messaging quality (e.g., timeliness of data delivery) in CPS sensornets. The stringent application requirements in CPS make it necessary to rethink about sensornet design, and one such problem is in-network processing.

For resource constrained sensornets, in-network processing (INP) improves energy efficiency and data delivery performance by reducing network traffic load and thus channel contention. Over the past years, many INP methods have been proposed for query processing [54, 69, 58, 15] and general data collection [20, 21, 43, 52, 61, 71]. Not focusing on mission-critical WCPS, however, these works have mostly ignored the Quality of Services constraints when designing INP mechanisms. Thus, the interaction between specific, real-world INP methods and QoS of data delivery remains a largely unexplored issue in WCPS systems. This is an important issue because

1. It affects the efficiency and quality of real-time, efficient and resilient embedded sensing and control;

2. As we will show later in this dissertation, different INP methods and their different constraints (e.g., aggregation capacity limit and re-aggregation tolerance) affect, to a greater extent than network and traffic properties, the complexity and the protocol design in jointly optimizing in-network processing and QoS of data delivery.

In this dissertation, we focus on two widely used INP methods, packet packing and network coding (which we use NC interchangeably hereafter), and their quality of services in mission-critical wireless networked sensing and control. Our results show that these two techniques can significantly improve network performance in terms of timeliness of data delivery, energy efficiency, delivery reliability and network throughput under stringent application QoS requirements in WCPS. More specifically, we study the joint optimization problem of packet packing and real-time constraints of data delivery, the minimal cost network-coding-based (NC-based) routing problem, and proactive NC- based protection problem for mission-critical WCPS.

Contribution of this dissertation

Before presenting all the details, we first summarize the contribution of this dissertation as follows:

1. We examine the complexity and impact of jointly optimizing packet packing and the timeliness of data delivery. We find that different packing constraints have a large effect on the problem complexity. We identify conditions for the joint optimization to be strong NP-hard and conditions for it to be solvable in polynomial time. We also develop a local, distributed online protocol *tPack* for maximizing the local utility of each node, and we prove the competitiveness of the protocol with respect to optimal solutions. Our measurement study on the NetEye testbed demonstrates the importance of QoS- and aggregation constraint aware optimization of packet packing.

2. We study the transmission cost minimization problem of network coding based routing. We propose the first mathematical framework to compute the transmission cost of NC-based routing. We then find that this minimization problem is polynomial solvable and designed an greedy optimal algorithm. We prove the optimality of this algorithm and conduct a theoretical comparison between our minimal cost NC-based routing and traditional single path routing. We show that not only the shortest single path routing is not necessarily selected into the optimal routing braid, but also that the optimal routing braid has a transmission cost upper bounded by the shortest single path routing. We develop a distributed NC-based routing protocol *EENCR* to implement this optimal algorithm.

EENCR inherits the advantages of both single path routing protocol and network coding based opportunistic routing protocol. Our measurement study on the NetEye in a new environment show that EENCR outperforms a state-of-the art single path routing protocol and two other classic network coding based opportunistic routing protocols in terms of reliability, delivery cost and goodput.

3. Based on our findings in the minimal cost NC-based routing, we also study the 1+1 proactive network coding based protection problem. We prove that finding 2 node-disjoint routing braids with minimal transmission cost for NC-based transmission is NP-hard even under a simple setting. We then propose a heuristic yet efficient algorithm to construct 2 node-disjoint routing braids by fully exploring the routing diversity in the network. We develop a proactive network coding protection protocol *ProNCP* to evaluate this algorithm. Experiment results show that ProNCP is resilient to various random transient node failures in wireless networked sensing and control systems by providing a close to 100% delivery reliability and incurring only a 50% transmission cost compared with the classic 2 shortest node-disjoint path algorithm.

Organization of this dissertation

The rest of this dissertation is organized as follows. In Chapter 2 we present our study on joint optimization between packet packing and real-time constraints of data delivery. In Chapter 3, we study the energy-efficient NC-based routing problem. We then study the 1+1 proactive NC-based protection problem in Chapter 4. We conclude this dissertation in Chapter 5.

CHAPTER 2
REAL-TIME PACKET PACKING SCHEDULING

Preliminary

Towards understanding the interaction between INP and data delivery latency in foreseeable real-world sensornet deployments, we focus on a widely used, application-independent INP method — *packet packing* where multiple short packets are aggregated into a single long packet [29, 53]. In sensornets (especially those for real-time sensing and control), an information element from each sensor is usually short, for instance, less than 10 bytes [9, 54]. Yet the header overhead of each packet is relatively high in most sensornet platforms, for instance, up to 31 bytes at the MAC layer alone in IEEE 802.15.4 based networks. It is also expected that more header overhead will be introduced at other layers (e.g., routing layer) as we standardize sensornet protocols such as in the effort of the IETF working groups 6LowPAN [4] and ROLL [33]. Besides header overhead, MAC coordination also introduces non-negligible overhead in wireless networks [53]. If we only transmit one short information element in each packet transmission, the high overhead in packet transmission will significantly reduce the network throughput; this is especially the case for high speed wireless networks such as IEEE 802.15.4a ultrawideband (UWB) networks. Fortunately, the maximum size of packet payload is usually much longer than that of each information element, for instance, 128 bytes per MAC frame in 802.15.4. Therefore, we can aggregate multiple information elements into a single packet to reduce the amortized overhead of transmitting each element. Packet packing also reduces the number of packets contending for channel access, hence it reduces the probability of packet collision and improves information delivery reliability, as we will show in Chapter . The benefits of packet packing have also been recognized by the IETF working groups 6LowPAN and ROLL.

Unlike total aggregation assumed in [10] and [59], the number of information elements that can be aggregated into a single packet is constrained by the maximum packet size, thus we have to carefully schedule information element transmissions so that the degree of packet packing (i.e., the amount of sensing data contained in packets) can be maximized without violating application requirement on the timeliness of data delivery. As a first step toward understanding the complexity of jointly optimizing INP and QoS with aggregation constraints, we analyze the impact that aggregation constraints have on the computational complexity of the problem, and we prove the following:

1. When a packet can aggregate three or more information elements, the problem is strong NP-hard, and there is no polynomial-time approximation scheme (PTAS).

2. When a packet can only aggregate two information elements, the complexity depends on whether two elements in a packet can be separated and re-packed with other elements on their way to the sink: if the elements in a packet can be separated, the problem is strong NP-hard and there is no PTAS for the problem; otherwise it can be solved in polynomial time by modeling the problem as a maximum weighted matching problem in an interval graph.

3. The above conclusions hold whether or not the routing structure is a tree or a linear chain, and whether or not the information elements are of equal length.

Besides shedding light on the complexity and protocol design of jointly optimizing data delivery timeliness and packet packing (as well as other INP methods), these findings incidentally answer several open questions on the complexity of batch-process scheduling in interval graphs [22].

To understand the impact of jointly optimizing packet packing and data delivery timeliness, we design a distributed, online protocol *tPack* that schedules packet transmissions to maximize the local utility of packet packing at each node while taking into account the aggregation constraint imposed by the maximum packet size. Using a testbed of 130 TelosB motes, we experimentally evaluate the properties of tPack. We find that jointly optimizing data delivery timeliness and packet packing and considering real-world aggregation constraints significantly improve network performance (e.g. in terms of high reliability, high energy efficiency, and low delay jitter).

Table 1: Notations used in Chapter 2

Common notations	
K	maximum number of information elements allowed in a packet
$ETX_{v_i v_j}(l)$	expected number of transmissions taken to successfully deliver a packet of length l along link (v_i, v_j)
$t_{v_i v_k}(l)$	maximum time taken to deliver a packet of length l from v_i to v_k in the absence of packet packing and packing-oriented scheduling
Notations used in the section of preliminary study only	
R	root of a directed collection tree
x	an information element
v_x	the node where x is generated
l_x	length of x
r_x	time when x is generated
d_x	deadline of delivering x to R
s_x	spare time in delivering x
$[r_x, d_x]$	lifetime of x
Notations used in the section of complexity study only	
n	number of variables in a SAT instance
m	number of clauses in a SAT instance
X_j	jth variable of a SAT instance
C_i	ith clause of a SAT instance
x_i^j	information element corresponding to the variable X_j in a clause C_i
$[r_i^j, d_i^j]$	lifetime of x_i^j
ax_k^j	kth auxiliary information element for variable X_j
$[r_{ax_k}^j, d_{ax_k}^j]$	lifetime of ax_k^j
z_i	information element generated by node v_i^c
$[r_i, d_i]$	lifetime of z_i
t_1	transmission time from any leaf node to its parent
t_2	transmission time from any node v_j to node v
t_3	transmission time from node v to node s
t_4	transmission time from any node v_i^c to node v

The rest of this chapter is organized as follows. We first analyze the benefits of packet packing in lossy wireless networks in We then discuss the system model and precisely define the joint optimization problem in. Next we analyze the complexity of the problem in under different settings, and present the tPack protocol to provide a distributed solution to this problem. After presenting the protocol design and implementation details, we experimentally evaluate the performance of tPack and study the impact of packet packing as well as joint optimization. We discuss related work before concluding this study in the end of this chapter. For convenience, we summarize in Table 1 the notations used in the section of preliminary study and the section of complexity study.

Motivation for packing

While aggregating short information elements reduces the overhead of transmitting each information element, it increases the length of packets being transmitted. Given that packet delivery rate of a wireless link decreases as packet length increases, a longer packet with aggregated information elements may be retransmitted more often, for reliable data delivery, than the shorter packets without aggregation. To understand whether packet packing is still beneficial in the presence of lossy wireless links, therefore, we need to understand whether the increased packet loss rate overshadows the benefits of packet packing. To this end, we mathematically analyze the issue as follows.

For simplicity, we assume in this section that the status (i.e., success or failure) of different packet transmissions are independent, and we corroborate the analytical results through testbed based measurement in later sections where temporal link correlation exists. For convenience, we define the following notations:

l_1 : payload length of an unpacked packet, i.e., the length of a single information element;

p_1 : delivery rate of an unpacked packet;

k : packing ratio, i.e., the ratio of the payload length of a packed packet to that of an unpacked packet;

h : the ratio of header length to payload length in an unpacked packet.

Then, for a packed packet with packing ratio k, the ratio of the overall length of the packed packet to that of an unpacked packet is $\frac{kl_1+hl_1}{l_1+hl_1}$. Thus, the delivery rate p_k of the packed packet can be calculated as follows:

$$p_k = p_1^{\frac{kl_1+hl_1}{l_1+hl_1}} = p_1^{\frac{k+h}{1+h}}$$

To reflect the overhead of transmitting a packet pkt over a wireless link, we define the *amortized cost* (AC) of transmitting pkt as follows:

$$AC_{pkt} = \frac{ETX_{pkt}}{len_{pkt}} \tag{1}$$

where len_{pkt} is the payload length of pkt, and ETX_{pkt} is the expected number of transmissions taken to successfully deliver pkt over the wireless link. Given that the expected number of transmissions to successfully deliver a packet with packing ratio k is $\frac{1}{p_k}$, the amortized cost of transmitting a packet with packing ratio k, denoted by AC_k, can be calculated as follows:

$$AC_k = \frac{1/p_k}{kl_1} = \frac{1}{kl_1p_k} \tag{2}$$

Since an unpacked packet has a packing ratio of 1, the amortized cost of transmitting an unpacked packet is AC_1, that is, $\frac{1}{l_1p_1}$.

For a given packing ratio k, the ratio R_k of AC_1 to AC_k reflects whether packet packing is beneficial, that is, packet packing is beneficial if $R_k > 1$. Precisely, R_k is calculated as follows:

$$R_k = \frac{AC_1}{AC_k} = kp_1^{\frac{k-1}{1+h}} \tag{3}$$

In a typical sensornet system [9, 8], the ratio h of header length to that of a single information element is around 3, and the packing ratio can be up to 12. For $h = 3$, Figure 1 shows R_k as a function of p_1 and k, when $h = 3$. From the figure, we can see that packet packing reduces the amortized cost of packet transmission as long as the link reliability is no less than 40%, which is usually the case in practice (e.g., link reliability was ∼75% even in heavily loaded sensornet systems [9, 8]). We also see that, if link reliability is greater than 67%, the amortized

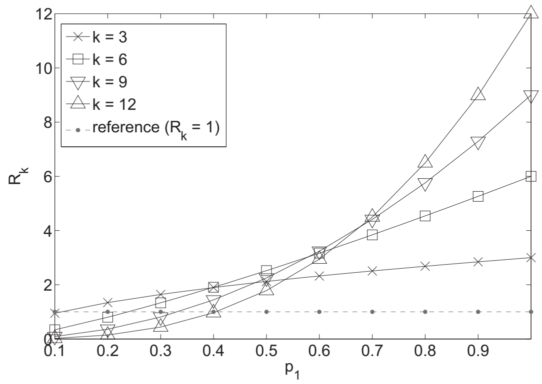

Figure 1: $R_k = \frac{AC_1}{AC_k}$

cost of packet transmission always decreases as the packing ratio increases. Since link reliability is usually greater than 67% in practice, we can always try to maximize the packing ratio so that the amortized cost of packet transmission is reduced.

Denoting k^* as the optimal packing ratio that minimize the amortized cost for transmitting a packet, we then study the relationship between k^* and p_1. From Equation 2, we have:

$$AC_k = \frac{1}{kl_1 p_k} = \frac{1}{kl_1 p_1^{\frac{k+h}{1+h}}} \tag{4}$$

To minimize AC_k, we need to maximize $f(k) = kl_1 p_1^{\frac{k+h}{1+h}}$. When $k \in R^+$, $f(k)$ is a convex function. Let $f'(k) = 0$. we have $k_R^* = \frac{1+h}{\ln p_1^{-1}}$. Therefore, when $k \in N^+$, k^* is calculated as follows:

$$k^* = \arg min_{k \in \{1, \lceil k_R^* \rceil, \lfloor k_R^* \rfloor\}} \{AC_k\} \tag{5}$$

In Figure 2, k^* increases as the link reliability increases. When p_1 is greater than 75%, k^* increase faster, which implies that packet packing can bring more benefit on amortized cost when link reliability is high. Figure 3 shows the relationship between AC_k and k when $p_1 = 0.9$. From the figure we can find that it is not always beneficial to pack as many small packets as possible. There exists a threshold on the packing ratio. When k exceeds this threshold, the amortized cost will increase. This motivates us to explore how to perform packing at each node

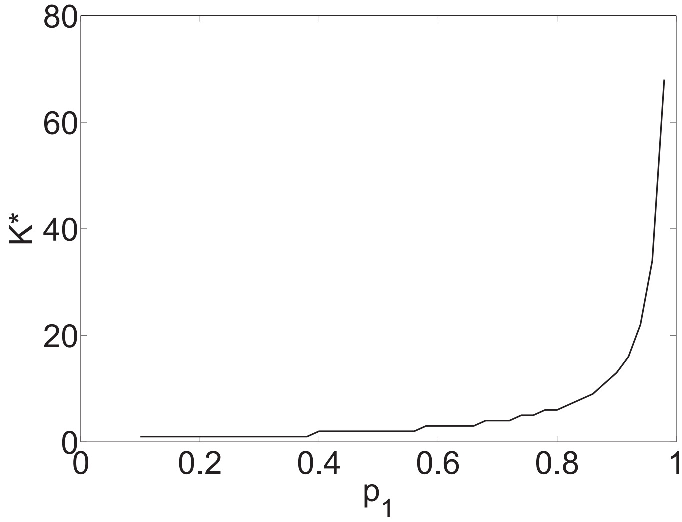

Figure 2: K^* when $l_1 = 12$, $h = 0.375$, and $K_{max} = 100$.

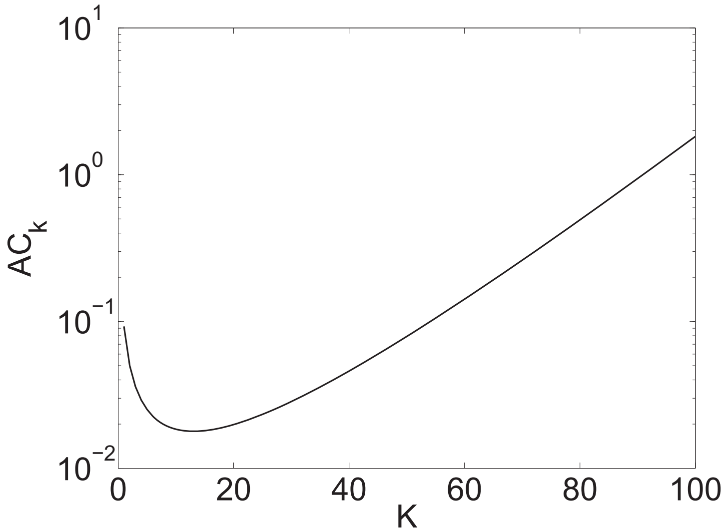

Figure 3: AC_K when $l_1 = 12$, $h = 0.375$, and $K_{max} = 100$.

in the network.

Remarks: The above analysis focuses on a single link, but the observations easily carry over to multi-hop networks since link reliability p_1 reflects the impact of channel fading and collision even in the case of multi-hop networks.[1] The analysis has not considered the benefits (e.g., fewer number of packet collisions) of reduced channel contention as a result of packet packing (which reduces the number of packets contending for channel access). We will study the impact of these factors through testbed based measurement in the performance evaluation section.

System model and problem definition

Having verified the benefits of packet packing in lossy wireless networks in last section, we now discuss the system model and define the joint optimization problems we will focus on in this paper.

System Model

We consider a directed collection tree $T = (V, E)$, where V and E are the set of nodes and edges in the tree. $V = \{v_i : i = 1 \dots N\} \cup \{R\}$ where R is the root of the tree and represents the data sink of a sensornet, and $\{v_i : i = 1 \dots N\}$ are the set of N sensor nodes in the network. An edge $\langle v_i, v_j \rangle \in E$ if v_j is the parent of v_i in the collection tree. The parent of a node v_i in T is denoted as p_i. We use $ETX_{v_i v_j}(l)$ to denote the expected number of transmissions required for delivering a packet of length l from a node v_i to its ancestor v_j, and we use $t_{v_i v_k}(l)$ to denote the maximum time taken to deliver a packet of length l from v_i to v_k in the absence of packet packing and packing-oriented scheduling.

Each information element x generated in the tree is identified by a 4-tuple (v_x, l_x, r_x, d_x) where v_x is the node that generates x, l_x is the length of x, r_x is the time when x is generated, and d_x is the deadline by which x needs to be delivered to the sink node R. We use $s_x =$

[1]Note that the increased per-packet transmission time as a result of increased packet length will not cause more collision, since the time taken to transmit a packet (e.g., $\sim$ 4 milliseconds) is usually much less than the inter-packet interval (e.g., usually at least a few seconds).

$d_x - (r_x + t_{v_x R}(l_x))$ to denote the *spare time* for x, and we define the *lifetime of* x as $[r_x, d_x]$.

Problem Definition

Given a collection tree T and a set of information elements $X = \{x\}$ generated in the tree, we define the problem of jointly optimizing packet packing and the timeliness of data delivery as follows:

Problem $\mathbb{P}$: given T and X, schedule the transmission of each element in X to minimize the total number of packet transmissions required for delivering X to the sink R while ensuring that each element be delivered to R before its deadline.

In an application-specific sensornet, the information elements generated by different nodes depend on the application but may well be of equal length [9]. Depending on whether the sensornet is designed for event detection or data collection, moreover, the information elements X may follow certain arrival processes. Based on the specific arrival process of X, the following special cases of problem $\mathbb{P}$ tend to be of practical relevance in particular:

Problem $\mathbb{P}_0$: same as $\mathbb{P}$ except that 1) the elements of X are of equal length, and 2) X includes at most one element from each node; this problem can represent sensornets that detect rare events.

Problem $\mathbb{P}_1$: same as $\mathbb{P}$ except that 1) the elements of X are of equal length, and 2) every two consecutive elements generated by the same node v_i are separated by a time interval whose length is randomly distributed in $[a, b]$; this problem can represent periodic data collection sensornets (with possible random perturbation to the period).

Problem $\mathbb{P}_2$: same as $\mathbb{P}$ except that the elements of X are of equal length; this problem represents general application-specific sensornets.

Complexity of joint optimization

The complexity of problem $\mathbb{P}$ depends on aggregation constraints such as maximum packet size and whether information elements in a packet can be separated and repacked with other elements. For convenience, we use K to denote the maximum number of information elements that can be packed into a single packet. (Note that K depends on the maximum packet size and the lengths of information elements in problem $\mathbb{P}$.) In what follows, we first analyze the case when $K \geq 3$ and then the case when $K = 2$, and we discuss how aggregation constraints affect the problem complexity.

Complexity when $K \geq 3$

We first analyze the complexity and the hardness of approximation for problem $\mathbb{P}_0$, then we derive the complexity of $\mathbb{P}_1$, $\mathbb{P}_2$, and $\mathbb{P}$ accordingly. The analysis is based on reducing the Boolean-satisfiability-problem (SAT) [26] to $\mathbb{P}_0$ as we show below.

Theorem 1 *When $K \geq 3$, problem $\mathbb{P}_0$ is strong NP-hard whether or not the routing structure is a tree or a linear chain.*

Proof To prove that $\mathbb{P}_0$ is strong NP-hard, we first present a polynomial transformation f from the SAT problem to $\mathbb{P}_0$, then we prove that an instance Π of SAT is satisfiable if and only if the optimal solution of $\Pi' = f(\Pi)$ has certain minimum number of transmissions.

Given an instance Π of the SAT problem which has n Boolean variables $X_1, \ldots, X_n$ and m clauses $C_1, \ldots, C_m$, we derive a polynomial time transformation from Π to an instance Π' of $\mathbb{P}_0$ with $K \geq 3$ as follows. Firstly, we construct a tree with n+2 nodes shown in Figure 4. In this tree, each node v_j, where $j = 1, \ldots, n$ corresponds to the variable X_j. Node v is an

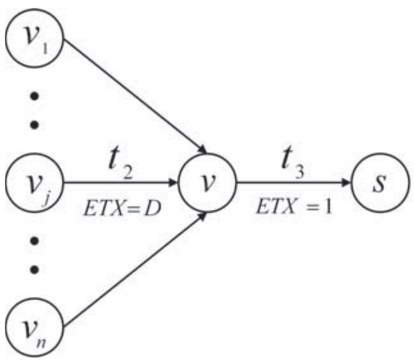

Figure 4: A tree with $n + 2$ nodes

intermediate node and node S is the base station. $ETX_{v_j v}$ is D, where $D \gg 1$, and ETX_{vs} is

1. (For now, we do not consider the impact of packet length on link reliability and thus ETX.) The transmission time $t_{v_j v} = t_2$ and $t_{vs} = t_3$. This operation takes $O(n)$ time.

Secondly, assume that variable X_j appears k_j times in total in the m clauses. Then we add $2k_j + 3$ children to node v_j, labeled as $v_0^j, \ldots, v_{2k_j+2}^j$, and m children to node v, labeled as $v_1^c, \ldots, v_m^c$. Each new edge has a ETX of 1. The transmission time from each child of v_j to v_j is t_1, and the transmission time from v_i^c to v is t_4. This operation takes $O(nm)$ time and the whole tree is shown in Figure 5.

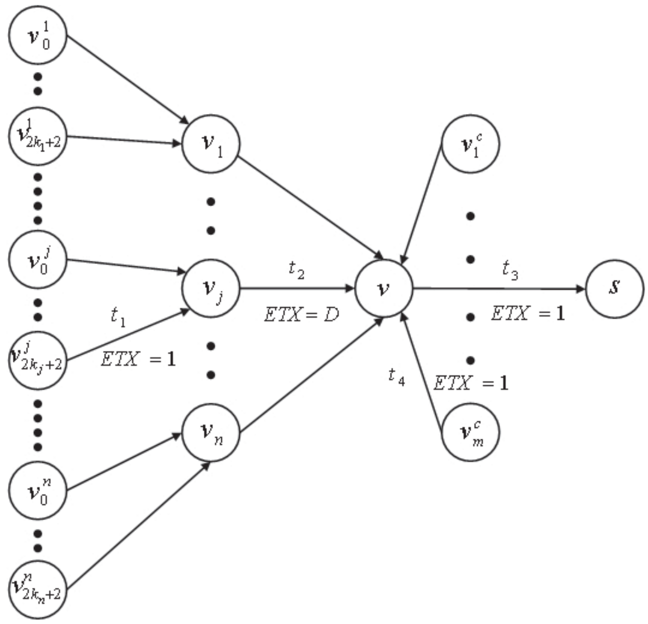

Figure 5: Reduction from SAT to $\mathbb{P}_0$ when $K \geq 3$

After constructing the tree, we define the information elements and their lifetimes as follows. For each subtree rooted at node v_j, we first define $2k_j + 1$ information elements and then assign them one by one to the leaf nodes $v_1^j, \ldots, v_{2k_j+1}^j$ of this subtree. If variable X_j occurs unnegated in clause C_i, we create an information element x_i^j with lifetime $[r_i^j, d_i^j] = [(3i+1)(n+1)+j, (3i+2)(n+1)+j+t_1+t_2+t_3]$. If X_j occurs negated in clause C_i, we create an information element $x_i^j : [r_i^j, d_i^j] = [3i(n+1)+j, (3i+1)(n+1)+j+t_1+t_2+t_3]$. Let $i_1^j < \ldots < i_{k_j}^j$ denote the indices of the clauses in which variable X_j occurs. For every two messages $x_{i_t^j}^j$ and $x_{i_{t+1}^j}^j$, $t = 1, \ldots, k_j - 1$, define an information element $ax_{i_t^j}^j : [r_{at}^j, d_{at}^j] = [d_{i_t^j}^j - t_1 - t_2 - t_3, r_{i_{t+1}^j}^j + t_1 + t_2 + t_3]$. We also define $ax_0^j : [r_{a0}^j, d_{a0}^j] = [j, r_{i_1^j}^j + t_1 + t_2 + t_3]$, and $ax^j_{k_j} : [r^j_{a_{k_j}}, d^j_{a_{k_j}}] = [d^j_{i^j_{k_j}} - t_1 - t_2 - t_3, 3(m+1)(n+1) + j + t_1 + t_2 + t_3]$. In this way, every two consecutive information elements in this sequence overlap in their lifetimes, and the size of the overlap is $t_1 + t_2 + t_3$. After defining these $2k_j + 1$ information elements, we set the source of each element one by one from node v^j_1 to node $v^j_{2k_j+1}$. For each node v^j_0, we define an element $z^j_0 : [j, j + t_1 + t_2 + t_3]$. For each node $v^j_{2k_j+2}$, we define an element $z^j_{2k_j+2} : [3(m+1)(n+1)+j, 3(m+1)(n+1)+j+t_1+t_2+t_3]$. Figure 6 demonstrates how

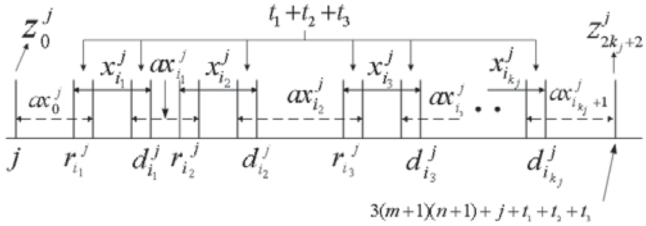

Figure 6: Lifetimes of information elements

the lifetimes of these $2k_j + 3$ information elements are defined.

Similarly, we define m information elements generated by nodes $v^c_1, \ldots, v^c_m$, with element $z_i : [r_i, d_i] = [(3i+1)(n+1)+t_1+t_2-t_4, (3i+2)(n+1)+t_1+t_2+t_3]$, $i = 1, \ldots, m$, being generated by node v^c_i. Then, for nodes v_1 to v_n, we define an information element for each of them with lifetime $[4(m+1)(n+1)+i, 4(m+1)(n+1)+i+t_2+t_3]$, $i = 1, \ldots, n$. For node v, define an information element with lifetime $[4(m+1)^2(n+1)+i, 4(m+1)^2(n+1)+i+t_3]$.

The whole process to assign an information element for each sensor will take $O(nm)$ time. Therefore, the time complexity of the whole transformation is $O(n) + O(nm) + O(nm) = O(nm)$, which is polynomial in n and m.

Given the instance Π' of $\mathbb{P}_0$ formulated as above, the following claims hold for the optimal packing scheme:

Claim 1 *If nodes $v^c_1, \ldots, v^c_m$ are ignored, the minimum total number of transmission in Π' is $C_{t0} = \sum_{j=1}^{n}(2k_j + 1) + \sum_{j=1}^{n}[(k_j + 1)(D + 1)] + 2n(D + 1) + 2n + 1$.*

Proof It is easy to see that the information elements generated by $v_i, i = 1, \ldots, n$, and v, cannot be packed with any other information elements. Therefore, the total number of transmission for these elements is $C_{t0}^1 = n(D+1)+1$.

Since the ETX of each link from v_j to v, $j = 1,\ldots,n$ is D and $D \gg 1$, and each sensor only generates one piece of information element, in an optimal packing scheme, every information element generated by node $v_{t_j}^j$, $t_j = 1,,2k_j+1$, will leave its source immediately it is generated and then seek the opportunity to pack with other information elements before it is forwarded from v_j to v. Due to our definition on lifetimes for every $2k_j+1$ elements generated by nodes $v_{t_j}^j, t_j = 1,\ldots,2k_j+1$, only at most two consecutive information elements in this $2k_j+1$ sequence can be packed together at node v_j. For any two consecutive information elements that are packed together, the first element, which is generated by $v_{t_j}^j$ leaves node v_j at time $d_{t_j}^j - (t_2+t_3)$, and the second element, which is generated by $v_{t_j+1}^j$ leaves node v_j at time $r_{t_j+1}^j + t_1$. Thus in an optimal packing scheme, for all $2k_j+1$ incoming elements, node v_j will pack them into at least k_j+1 packets, k_j of which contain two element. In each $2k_j+1$ sequence, either information element ax_o^j arrives at and leaves node v_j at time $j+t_1$ alone, or information element $ax_{k_j}^j$ arrives at and leaves node v_j at time $3(m+1)(n+1)+j+t_1$ alone. Thus, the total number of transmission for these elements is $C_{t0}^2 = \sum_{j=1}^n (2k_j+1) + \sum_{j=1}^n [(k_j+1)(D+1)]$.

Besides, we have $2n$ more information elements z_0^j and z_{m+1}^j, $j = 1,\ldots,n$, left. Due to the definition of lifetimes for these information elements, all of them need to leave their sources as soon as they are generated, and none of them can be packed with a packet containing two information elements we packed in the last paragraph. In an optimal packing scheme, for a fixed j, either z_0^j is packed with ax_0^j at node v_j, i.e., ax_0^j arrives at and leaves node v_j at time $j+t_1$, or z_{m+1}^j is packed with $ax_{k_j}^j$ at node v_j, i.e., $ax_{k_j}^j$ arrives at and leaves node v_j at time $3(m+1)(n+1)+j+t_1$, which is shown in Figure 7. Thus, the total number of transmission for these elements is $C_{t0}^3 = 2n + n(D+1)$. Under this packing scheme, no packet will contain more than 2 elements, which also satisfies the packing size constraint. Thus, the minimal total number of transmissions in this tree is $C_{t0}^1 + C_{t0}^2 + C_{t0}^3 = n(D+1)+1+\sum_{j=1}^n (2k_j+1) + \sum_{j=1}^n [(k_j+1)(D+1)] + 2n + n(D+1) = C_{t0}$. ∎

Claim 2 *If nodes $v_1^c,\ldots,v_m^c$ are ignored, in the optimal packing scheme in Π', every information element q generated by a leaf node of node $v_j, j = 1,\ldots,n$, is forwarded to the source's*

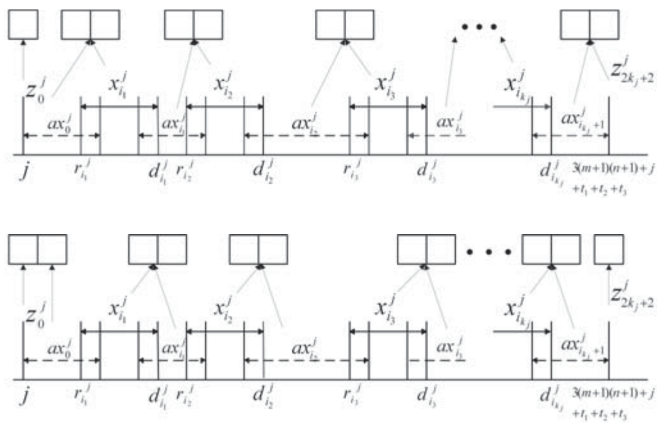

Figure 7: Example of optimal packing when $K \geq 3$

parent at time r_q, and then leaves the parent to next hop either at time $r_q + t_1$ or at time $d_q - (t_2 + t_3)$.

Proof Correctness of this claim can be easily verified by the definition of the information elements of those leaf nodes. ∎

Claim 3 *If nodes $v_1^c, \ldots, v_m^c$ are ignored, in the optimal packing scheme in Π', for each $j = 1, \ldots, n$, all the information elements x_i^j leaves node v_j for v either at time $r_i^j + t_1$, or at the time $d_i^j - (t_2 + t_3)$.*

Proof Since in an optimal packing scheme, either element z_0^j is packed with element ax_0^j, or element $z_{2k_j+2}^j$ is packed with element $ax_{k_j}^j$. If z_0^j is packed with ax_0^j, ax_0^j leaves v_j as soon as it arrives at v_j, when z_0^j arrives at v_j, i.e., each element $x_{i_t^j}^j$ leaves from v_j for v at time $d_{i_t^j}^j - (t_2 + t_3)$, packed with element $ax_{i_t^j}^j$, $t = 1, \ldots, k_j$. If $z_{2k_j+2}^j$ is packed with $ax_{k_j}^j$, $ax_{k_j}^j$ leaves v_j at time $3(m+1)(n+1)+j+t_1$, which equals to $d_{a_{k_j}0}^j - (t_2+t_3)$, when $z_{2k_j+2}^j$ arrives at v_j, i.e., each element $x_{i_t^j}^j$ leaves from v_j for v at time $r_{i_t^j}^j + t_1$, packed with element $ax_{i_t^j-1}^j$, $t = 1, \ldots, k_j$. ∎

From Claim 1, 2 and 3, we present the following claim:

Claim 4 *The minimum number of transmissions required in Π', denoted by C_{t1}, is $C_{t0} + m$ if and only if the SAT problem Π is satisfiable.*

Proof 1) Given a satisfying assignment for the SAT problem, an optimal packing scheme of the corresponding packet packing problem can be derived as follows: If in the assignment of SAT problem, variable X_j is set true, then all information elements x_i^j are forwarded from their sources to node v_j at time r_i^j, and are forwarded from node v_j to node v at time $r_i^j + t_1$. If X_j is set false, then all information elements x_i^j are forwarded from their sources to node v_j at time r_i^j, and are forwarded from node v_j to node v at $d_i^j - t_2 - t_3$. Similarly with the information elements generated by children nodes of node v_j, every element generated by node v_i^c, $i = 1, \ldots, m$, cannot get packed at its source since v_i^c is a leaf node. As a result, each element z_i is forward by its source and arrives at node v at time $(3i+1)(n+1)+t_1+t_2-t_4+t_4 = (3i+1)(n+1)+t_1+t_2$. Then the spare period for information element z_i to wait at node v is $[(3i+1)(n+1)+t_1+t_2, (3i+2)(n+1)+t_1+t_2]$. If clause C_i is satisfied by setting X_j to be true, then information element x_i^j arrives at node v at $(3i+1)(n+1)+t_1+t_2+j \in [(3i+1)(n+1)+t_1+t_2, (3i+2)(n+1)+t_1+t_2]$, which implies z_i can be packed with any packet containing information element x_i^j. Similarly, if clause C_i is satisfied by setting X_j to be false, then information element x_i^j arrives at node v at $(3i+1)(n+1)+t_1+t_2+j \in [(3i+1)(n+1)+t_1+t_2, (3i+2)(n+1)+t_1+t_2]$, which implies z_i can be packed with any packet containing information element x_i^j. Figure 8 gives an example on how to get the optimal

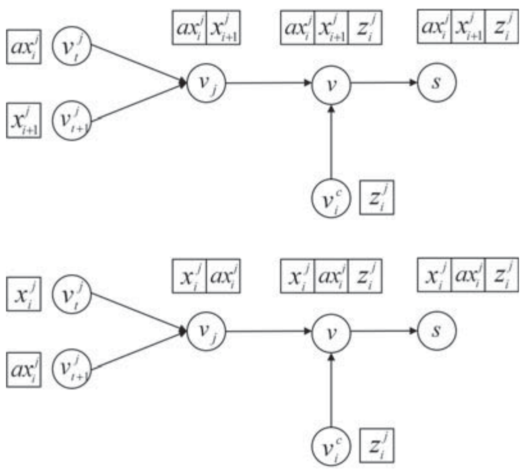

Figure 8: Deriving optimal packing scheme from SAT assignment when $K \geq 3$

packing scheme from an assignment of SAT instance.

Under this scheme, no packet will contain more than 3 elements, which also satisfies the packing size constraint. Every element z_i, $i = 1, \ldots, m$, can be packed at node v with a packet containing message x_i^j if clause C_i is satisfied due to variable X_j. Therefore, the additional number of transmission to send each element z_i to node s is m. As a result, the total number of transmission for this tree is $C_{t0} + m = C_{t1}$.

2) If we may find that the optimal packing scheme has a total number of transmission C_{t1}, which implies that every element z_i joins a packet consisting of x_i^j for some j value. If x_i^j leaves from node v_j at time $r_i^j + t_1$, and z_i joins the packet that contains x_i^j at node v, this can only happen when X_j is unnegated in clause C_i because $(3i+1)(n+1)+t_1+t_2+j \in [(3i+1)(n+1)+t_1+t_2, (3i+2)(n+1)+t_1+t_2]$ and $3i(n+1)+j \notin [(3i+1)(n+1)+t_1+t_2, (3i+2)(n+1)+t_1+t_2]$. Thus we set X_j to be true. If x_i^j leaves from node v at time $d_i^j - (t_2+t_3)$, and z_i joins the packet that contains x_i^j at node v, this can only happen when X_j is negated in clause C_i because $(3i+1)(n+1)+t_1+t_2+j \in [(3i+1)(n+1)+t_1+t_2, (3i+2)(n+1)+t_1+t_2]$ and $(3i+2)(n+1)+j+t_1+t_2 \notin [(3i+1)(n+1)+t_1+t_2, (3i+2)(n+1)+t_1+t_2]$. Thus we set X_j to be false. By this way, if we have an optimal solution to this instance of packet packing problem, we can have a satisfying assignment of the original SAT problem. Note that due to Claim 3, the following case cannot happen: element z_i gets packed with x_i^j by letting x_i^j leaves node v_j at time $r_i^j + t_1$, and in the meantime, that element z_k gets packed with x_k^j by letting x_k^j leaves node v_j at time $d_k^j - (t_2 + t_3)$. ∎

Then, Claim 4 and the fact that the reduction shown in Figure 5 is a polynomial reduction from SAT to $\mathbb{P}_0$ imply that $\mathbb{P}_0$ is strong NP-hard when $K \geq 3$. ∎

Note that the above proof did not consider the impact of packet length on link reliability and thus ETX. As long as we construct the reduction so that the ETX along links $\langle v_j, v\rangle, j = 1, \ldots, n$ is significantly greater than that along link $\langle v, s\rangle$, however, the above analysis can be easily extended to and still hold for cases where ETX is a function of packet length.

Having proved the strong NP-hardness of $\mathbb{P}_0$ when $K \geq 3$, we analyze the hardness of approximation for $\mathbb{P}_0$ using a gap-preserving reduction from MAX-3SAT to $\mathbb{P}_0$ [32], and we have

Theorem 2 *When $K \geq 3$, there exists $\epsilon \geq 1$ such that it is NP-hard to achieve an approximation ratio of $1 + \frac{1}{200N}(1 - \frac{1}{\epsilon})$ for problem $\mathbb{P}_0$, where N is the number of information elements in $\mathbb{P}_0$.*

Proof We first show that the reduction presented in Figure 5 is a gap-preserving reduction [32] from MAX-3SAT to problem $\mathbb{P}_0$. It is easy to verify that the proof of Theorem 1 holds if the discussion of the proof is based on 3SAT instead of the general SAT problem, in which case $\sum_{j=1}^{n} k_j = 3m$ and we denote the reduction as f. Therefore, if a 3SAT problem Π is satisfiable, the minimum cost of the $\mathbb{P}_0$ problem $\Pi' = f(\Pi)$ is

$$\begin{aligned} C_{t1} &= C_{t0} + m \\ &= (\textstyle\sum_{j=1}^{n}(2k_j + 1) + \sum_{j=1}^{n}(k_j + 1)(D + 1) + \\ &\quad 2n(D + 1) + 2n + 1) + m \\ &= m(3D + 10) + n(3D + 6) + 1 \end{aligned} \tag{6}$$

Since $n < 4m$, (6) implies that

$$\begin{aligned} C_{t1} &< m(3D + 10) + n(3D + 10) \\ &< 5m(3D + 10) \end{aligned} \tag{7}$$

Note that the proof of Theorem 1 holds if $D = n + \sum_{j=1}^{n}(2k_j + 3) = 6m + n$, which is the number of information elements generated by the descendants of node v. Thus, (7) implies that

$$\begin{aligned} C_{t1} &< 5m(3(6m + n) + 10) \\ &= 5m(18m + 3n + 10) \\ &< 5m(18m + 3 \times 4m + 10) \\ &= 5m(30m + 10) \\ &< 5m(30m + 10m) \\ &= 200m^2 \end{aligned} \tag{8}$$

If only m_0 of the m clauses in Π are satisfiable, then the minimum cost in $\Pi' = f(\Pi)$ (with $K \geq 3$ is $C_{t1} + m - m_0$. This is because $(m - m_0)$ number of z_i's cannot be packed with any other packet and have to be sent from node v to s alone, which incurs an extra cost of 1 each. Accordingly, if less than m_0 of the m clauses in Π are satisfiable, then the minimum cost C' in $\Pi' = f(\Pi)$ is greater than $C_{t1} + m - m_0$. Letting $\epsilon = \frac{m}{m_0}$, (8) implies that

$$\begin{aligned} \frac{C'}{C_{t1}} &> \frac{C_{t1}+m-m_0}{C_{t1}} \\ &= \frac{C_{t1}+\epsilon m_0-m_0}{C_{t1}} \\ &= 1 + \frac{(\epsilon-1)m_0}{C_{t1}} \\ &> 1 + \frac{(\epsilon-1)m_0}{200m^2} \\ &= 1 + \frac{\epsilon-1}{200m}\frac{1}{\epsilon} \\ &= 1 + \frac{1}{200m}(1-\frac{1}{\epsilon}) \\ &\geq 1 + \frac{1}{200N}(1-\frac{1}{\epsilon}) \end{aligned} \tag{9}$$

where N is the number of non-sink nodes in the network and $N \geq m$.

Let $OPT(\Pi)$ and $OPT(\Pi')$ be the optima of a MAX-3SAT problem Π and the corresponding $\mathbb{P}_0$ problem $\Pi' = f(\Pi)$. Then the polynomial-time reduction f from MAX-3SAT to $\mathbb{P}_0$ satisfy the following properties:

$$\begin{aligned} OPT(\Pi) = 1 &\implies OPT(\Pi') = C_{t1} \\ OPT(\Pi) < \frac{1}{\epsilon} &\implies OPT(\Pi') > C_{t1}(1 + \frac{1}{200N}(1-\frac{1}{\epsilon})) \end{aligned} \tag{10}$$

From [32], we know that there exists a polynomial-time reduction f_1 from SAT to MAX-3SAT such that, for some fixed $\epsilon > 1$, reduction f_1 satisfies

$$\begin{aligned} I \in SAT &\implies \text{MAX-3SAT}(f_1(I)) = 1 \\ I \notin SAT &\implies \text{MAX-3SAT}(f_1(I)) < \frac{1}{\epsilon} \end{aligned} \tag{11}$$

Then, (10) and (11) imply the following:

$$\begin{aligned} I \in SAT &\implies OPT(f(f_1(I))) = C_{t1} \\ I \notin SAT &\implies OPT(f(f_1(I))) > C_{t1}(1 + \tfrac{1}{200N}(1 - \tfrac{1}{\epsilon})) \end{aligned} \tag{12}$$

Therefore, it is NP-hard to achieve an approximation ratio of $1 + \frac{1}{200N}(1 - \frac{1}{\epsilon})$ for problem $\mathbb{P}_0$.

Based on the definition of polynomial time approximation scheme (PTAS) and Theorem 2, we then have

Corollary 1 *There is no polynomial time approximation scheme (PTAS) for problem $\mathbb{P}_0$ when $K \geq 3$.*

Based on the findings for $\mathbb{P}_0$, we have

Theorem 3 *When $K \geq 3$, problems $\mathbb{P}_1$, $\mathbb{P}_2$, and $\mathbb{P}$ are strong NP-hard whether or not the routing structure is a tree or a linear chain, and there is no polynomial-time approximation scheme (PTAS) for solving them.*

Proof To prove the hardness results for $\mathbb{P}_1$, let's consider a special case Π_1 of $\mathbb{P}_1$ where 1) every node is generating information elements using the same period p_0 and the same spare time s_0 for information elements, 2) p_0 is significantly larger than s_0, and 3) p_0 is significantly larger than the latest time r_0 when a node generates its first information element such that the following holds: in the optimal packing scheme for Π_1, no two elements from the same node can be aggregated into the same packet, and the i-th information element from one node cannot be packed with the j-th element from another node unless $i = j$. It is easy to see that the special case Π_1 does exist by properly choosing the parameters p_0, s_0, and r_0. Therefore, solving Π_1 becomes the same as solving an instance Π_0 of $\mathbb{P}_0$ where the information elements consist of the first element from every node of Π_1. Therefore, $\mathbb{P}_1$ is at least as hard as $\mathbb{P}_0$. Since $\mathbb{P}_0$ is strong NP-hard, $\mathbb{P}_1$ is strong NP-hard, and the there is no PTAS for the problem.

Since $\mathbb{P}_1$ is a special case of $\mathbb{P}_2$, and $\mathbb{P}_2$ is a special case of $\mathbb{P}$, both $\mathbb{P}_2$ and $\mathbb{P}$ are strong NP-hard too, and there is no PTAS for them. ∎

Theorems 1 and 3 show that the joint optimization problems are strong NP-hard and there is no PTAS, whether or not the routing structure is a tree or a linear chain and whether or not the information elements are of equal length. In contrast, Becchetti $et\ al.$ [10] showed that, for total aggregation, the joint optimization problems are solvable in polynomial time via dynamic programming on chain networks. Therefore, we see that aggregation constraints make the difference on whether a problem is tractable for certain networks, and thus it is important to consider them in the joint optimization. Incidentally, we note that Theorem 3 also answers the open question on the complexity of Problem (P4) of batch-process scheduling in interval graphs [22].

Complexity when $K = 2$

We showed in the previous section that the problems $\mathbb{P}_i, i = 0, 1, 2$, and $\mathbb{P}$ are all strong NP-hard and there is no PTAS for these problems when $K \geq 3$. We prove in this section that, when $K = 2$, the complexity of these problems depends on whether information elements in a packet can be separated and re-packed with other elements (which we call *re-aggregation* hereafter) on their way to the sink. When re-aggregation is disallowed, these problems are solvable in polynomial time; otherwise they are strong NP-hard. Note that, when $K \geq 3$, these problems are all strong NP-hard even if re-aggregation is disallowed, which can be seen from the proof of Theorem 1. Note also that, even though re-aggregation may well be allowed in most sensornet systems when the in-network processing (INP) method is packet packing, re-aggregation may not be possible or allowed when INP is data fusion such as lossy data compression [67]. Via the study on the impact of re-aggregation, therefore, we hope to shed light on the structure of the joint optimization problems when general INP methods are considered.

In what follows, we first analyze the case when re-aggregation is allowed, then we analyze the case when re-aggregation is disallowed.

When re-aggregation is allowed

Use a method similar to that of Theorem1, we prove the following theorem.

Theorem 4 *When $K = 2$ and re-aggregation is allowed, problem $\mathbb{P}_0$ is strong NP-hard, and this result holds whether or not the routing structure is a tree or a linear chain.*

Proof Given an instance Π of SAT problem with n Boolean variables $X_1, \ldots, X_n$ and m clauses $C_1, \ldots, C_m$, we derive a polynomial time transformation from Π to an instance Π'' of problem $\mathbb{P}_0$ with $K = 2$ as follows. The transformation is the same as what we present through Figure 5 except for the following changes:

- Define a node p between node v and node s, and m children $p_1, \ldots, p_m$ of node p. Additionally, define $ETX_{vp} = ETX_{ps} = ETX_{p_ip} = 1$, and $t_{vp} = t_3, t_{ps} = t_5$, and $t_{p_ip} = t_6$.
- Define m information elements g_i's generated by nodes $p_1, , p_m$: $g_i : [r_i^p, d_i^p] = [(3i + 1)(n+1) + n + 0.1 + t_1 + t_2 + t_3 - t_6, (3i+1)(n+1) + n + 0.1 + t_1 + t_2 + t_3 + t_5]$, and for node p, define an information element g with lifetime $[5(m+1)^2(n+1) + i, 5(m + 1)^2(n+1) + i + t_5]$.
- For all parameters defined during the transformation in Figure 5, replace t_3 by $t_3 + t_5$.

Therefore, the time complexity of the new transformation is still $O(nm)$, and the new reduction is shown in Figure 9.

Then, the following claims hold for Π'':

Claim 5 *If nodes $v_1^c, \ldots, v_m^c$, and nodes $p_1, \ldots, p_m$ are ignored, the minimum number of transmissions in Π'' is $C'_{t0} = \sum_{j=1}^n (2k_j + 1) + \sum_{j=1}^n [(k_j + 1)(D + 2)] + 2n(D + 2) + 2n + 3$.*

Claim 6 *If nodes $v_1^c, \ldots, v_m^c$, and nodes $p_1, \ldots, p_m$ are ignored, in the optimal packing scheme of Π'', every information element q generated by a leaf node of node v_j,$j = 1, \ldots, n$, is forwarded to the source's parent at time r_q, and then leaves the parent to next hop either at time $r_q + t_1$, or at time $d_q - (t_2 + t_3 + t_5)$.*

Claim 7 *If nodes $v_1^c, \ldots, v_m^c$, and nodes $p_1, \ldots, p_m$ are ignored, in the optimal packing scheme of Π'', for each $j = 1, \ldots, n$, all the information elements x_i^j leave node v_j for v either at time $r_i^j + t_1$, or at time $d_i^j - (t_2 + t_3 + t_5)$.*

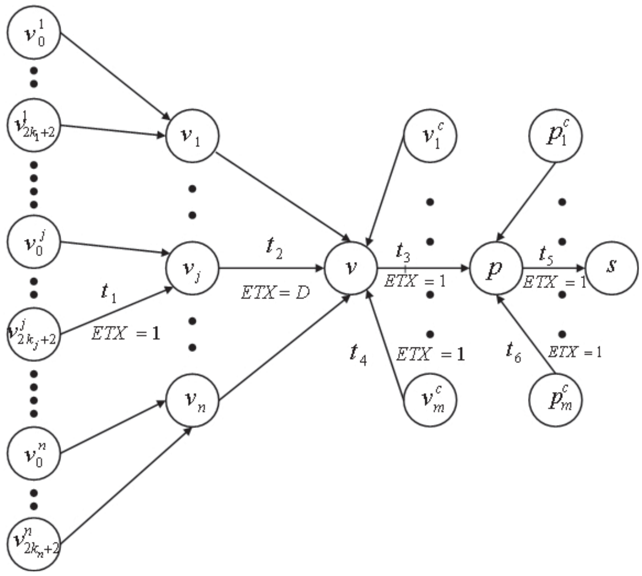

Figure 9: Reduction from SAT to $\mathbb{P}_0$ when $K = 2$

These claims can be proved in the same way as how Claims 1, 2, and 3 are proved respectively, and we skip the details here. Then, we propose

Claim 8 *The minimal number of transmissions required in* Π''*, denoted by* C'_{t1}*, is* $C'_{t0} + 4m$ *if and only if the SAT problem* Π *is satisfiable.*

Proof 1) Given a satisfying assignment for the SAT problem, an optimal packing scheme of the corresponding packet packing problem can be derived as follows: If in the assignment of SAT problem, variable X_j is set true, then all information elements x_i^j are forwarded from their sources to node v_j at time r_i^j, and are forwarded from node v_j to node v at time $r_i^j + t_1$. If X_j is set false, then all information elements x_i^j are forwarded from their sources to node v_j at time r_i^j, and are forwarded from node v_j to node v at $d_i^j - (t_2 + t_3 + t_5)$. Similarly with the information elements generated by children nodes of node v_j, every information element generated by node $v_i^c, i = 1, \ldots, m$, cannot get packed at its source since v_i^c is a leaf node. As a result, each information element z_i is forward by its source and arrives at node v at time $(3i + 1)(n + 1) + t_1 + t_2 - t_4 + t_4 = (3i + 1)(n + 1) + t_1 + t_2$. Then the spare period for information element z_i to wait at node v is $[(3i+1)(n+1)+t_1+t_2, (3i+2)(n+1)+t_1+t_2]$. If clause C_i is satisfied by setting X_j to be true, then information element x_i^j arrives at node v

at $(3i+1)(n+1)+t_1+t_2+j \in [(3i+1)(n+1)+t_1+t_2, (3i+2)(n+1)+t_1+t_2]$, which implies that z_i can be packed with the packet containing information element x_i^j. Similarly, if clause C_i is satisfied by setting X_j to be false, then information element x_i^j arrives at node v at $(3i+1)(n+1)+t_1+t_2+j \in [(3i+1)(n+1)+t_1+t_2, (3i+2)(n+1)+t_1+t_2]$, which implies z_i can be packed with the packet containing information element x_i^j. However, due to the packet size constraint, one packet cannot contain more than 2 information elements. In the meantime, every information element generated by node p_i cannot get packed at its source since node p_i is a leaf node. Thus each information element g_i is forwarded by its source and arrives at node p at time $(3i+1)(n+1)+n+0.1+t_1+t_2+t_3$. Then the spare period for element g_i to wait at node p is 0. In this case, to minimize the total number of transmission, if clause C_i is satisfied by setting X_j to be true, information element x_i^j arrives at node v with information element ax_{i-1}^j at time $(3i+1)(n+1)+t_1+t_2+j$ in one packet. When this packet arrives at v, information element ax_{i-1}^j and information element z_i form a new packet while information element x_i^j waits at v until $(3i+1)(n+1)+t_1+t_2+n+0.1$. x_i^j arrives at node g at time $(3i+1)(n+1)+n+0.1+t_1+t_2+t_3$ and forms a new packet with information element g_i. In this scheme, ax_{i-1}^j first packed x_i^j at node v_j, then leaves x_i^j at node v so that x_i^j can pack another information element g_i some time later at node p, which implies that a carry-over operation is used to achieve the optimal packing scheme. Similarly, if clause C_i is satisfied by setting X_j to be false, element x_i^j is arrives at node v with element ax_i^j at time $(3i+1)(n+1)+t_1+t_2+j$ in one packet. When this packet arrives at v, information element x_i^j and information element z_i form a new packet while information element ax_i^j waits at v until $(3i+1)(n+1)+t_1+t_2+n+0.1$. ax_i^j arrives at node p at time $(3i+1)(n+1)+n+0.1+t_1+t_2+t_3$ and forms a new packet with information element g_i. In this scheme, x_i^j first packed ax_i^j at node v_j, then leaves ax_i^j at node v so that ax_i^j can pack another information element g_i some time later at node p, which implies that a carry-over operation is used to achieve the optimal packing scheme. An demonstration on how the optimal packing scheme is derived is given in Figure 10.

In the optimal packing scheme, every information element z_i can be packed at node v with

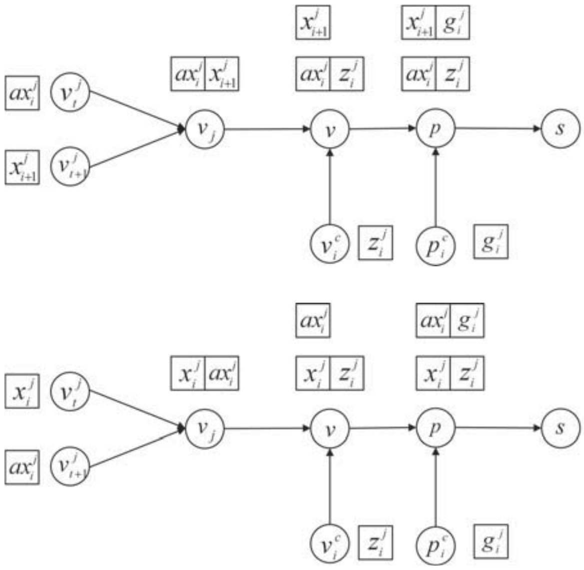

Figure 10: Deriving optimal packing scheme from SAT assignment when $K = 2$

an information element x_i^j or ax_{i-1}^j if clause C_i is satisfied due to variable X_j. Therefore, the additional number of transmission to send each information element z_i to node s is m, and the additional number of transmission to send each information element g_i to node s is m, and the additional number of transmission to break up m packet at node v and send them to node s is $2m$. As a result, the total number of transmission for this tree is $C'_{t0} + 4m = C'_{t1}$.

2) If we may find that the optimal packing scheme has a total number of transmission C'_{t1}, which implies that every information element z_i pack with one information element in a packet consisting of x_i^j for some j value, and the other information element in the old packet packs with information element g_i. If x_i^j leaves from node v_j at time $r_i^j + t_1$, and z_i packs with one information element in the packet that contains x_i^j at node v, this can only happen when X_j is unnegated in clause C_i because $(3i+1)(n+1)+t_1+t_2+j \in [(3i+1)(n+1)+t_1+t_2, (3i+2)(n+1)+t_1+t_2]$ and $3i(n+1)+j \notin [(3i+1)(n+1)+t_1+t_2, (3i+2)(n+1)+t_1+t_2]$. Thus we set X_j to be true. If x_i^j leaves from node v at time $d_i^j - (t_2+t_3+t_5)$, and z_i packs with one information element in the packet that contains x_i^j at node v, this can only happen when X_j is negated in clause C_i because $(3i+1)(n+1)+t_1+t_2+j \in [(3i+1)(n+1)+t_1+t_2, (3i+2)(n+1)+t_1+t_2]$ and $(3i+2)(n+1)+j+t_1+t_2 \notin [(3i+1)(n+1)+t_1+t_2, (3i+2)(n+1)+t_1+t_2]$. Thus we set X_j to be false. By this way, if we have an optimal solution to this instance of packet

packing problem, we can have a satisfying assignment of the original SAT problem. Note that due to Claim 7, the following case cannot happen: element z_i gets packed with x_i^j by letting x_i^j leaves node v_j at time $r_i^j + t_1$, and in the meantime, that element z_k gets packed with x_k^j by letting x_k^j leaves node v_j at time $d_k^j - (t_2 + t_3 + t_5)$. ∎

Then, Claim 8 and the fact that the reduction shown in Figure 9 is polynomial imply that $\mathbb{P}_0$ is strong NP-hard when $K = 2$. ∎

Note that the above proof did not consider the impact of packet length on link reliability and thus ETX. As long as we construct the reduction so that the ETX along links $\langle v_j, v\rangle, j = 1, \ldots, n$ is significantly greater than that along links $\langle v, p\rangle$ and $\langle p, s\rangle$, however, the above analysis can be easily extended to and still hold for cases where ETX is a function of packet length.

Note also that the above proof can be extended to the case when all the information elements are generated at the same time, as well as the case when the routing structure is a linear chain (with information elements having different generation time).

Then, we prove the hardness of approximation using a gap-preserving reduction from MAX-3SAT, and we have

Theorem 5 *When $K = 2$ and re-aggregation is allowed, there exists $\epsilon \geq 1$ such that it is NP-hard to achieve an approximation ratio of $1 + \frac{1}{120N}(1 - \frac{1}{\epsilon})$ for problem $\mathbb{P}_0$, where N is the number of information elements in $\mathbb{P}_0$.*

Proof The proof is similar to that of Theorem 2.

We first show that the reduction presented in Figure 9 is a gap-preserving reduction [32] from MAX-3SAT to problem $\mathbb{P}'_0$. It is easy to verify that the proof of Theorem 4 holds if the discussion of the proof is based 3SAT instead of the general SAT problem, in which case $\sum_{j=1}^{n} k_j = 3m$ and we denote the reduction as f. Therefore, if a 3SAT problem Π is satisfiable, the minimum cost of the $\mathbb{P}'_0$ problem $\Pi' = f(\Pi)$ is

$$\begin{aligned} C'_{t1} &= C'_{t0} + 4m \\ &= (\textstyle\sum_{j=1}^{n}(2k_j+1) + \sum_{j=1}^{n}(k_j+1)(D+2) + \\ &\quad 2n(D+2) + 2n + 3) + 4m \\ &= m(3D+16) + n(3D+9) + 3 \end{aligned} \tag{13}$$

Since $n < 4m$, Equation 13 implies that

$$\begin{aligned} C'_{t1} &< m(3D+16) + n(3D+16) \\ &< 5m(3D+16) \end{aligned} \tag{14}$$

Note that the proof of Theorem 4 holds if $D = n + \sum_{j=1}^{n}(2k_j+3) = 6m+n$, which is the number of information elements generated by the descendants of node v. Thus, Equation 14 implies that

$$\begin{aligned} C'_{t1} &< 5m(3(6m+n)+16) \\ &= 5m(18m+3n+16) \\ &< 5m(18m+3\times 4m+16) \\ &= 5m(30m+16) \\ &< 5m(30m+16m) \\ &= 240m^2 \end{aligned} \tag{15}$$

If only m_0 of the m clauses in Π are satisfiable, then the minimum cost in $\Pi' = f(\Pi)$ (with $K = 3$ is $C'_{t1} + (m - m_0)$. This is because $(m - m_0)$ number of z_i's cannot be packed with any other packet and have to be sent from node v to s alone, which incurs an extra cost of 2 each. Accordingly, if less than m_0 of the m clauses in Π are satisfiable, then the minimum cost C' in

$\Pi' = f(\Pi)$ is greater than $C'_{t1} + 2(m - m_0)$. Letting $\epsilon = \frac{m}{m_0}$, Equation 15 implies that

$$\begin{aligned}
\frac{C'}{C'_{t1}} &> \frac{C'_{t1}+2(m-m_0)}{C'_{t1}} \\
&= \frac{C'_{t1}+2(\epsilon m_0-m_0)}{C'_{t1}} \\
&= 1+2\frac{(\epsilon-1)m_0}{C'_{t1}} \\
&> 1+2\frac{(\epsilon-1)m_0}{240m^2} \\
&= 1+\frac{\epsilon-1}{120m}\frac{1}{\epsilon} \\
&= 1+\frac{1}{120m}(1-\frac{1}{\epsilon}) \\
&\geq 1+\frac{1}{120N}(1-\frac{1}{\epsilon})
\end{aligned} \tag{16}$$

where N is the number of non-sink nodes in the network and $N \geq m$.

Let $OPT(\Pi)$ and $OPT(\Pi')$ be the optima of a MAX-3SAT problem Π and the corresponding $\mathbb{P}'_0$ problem $\Pi' = f(\Pi)$. Then the polynomial-time reduction f from MAX-3SAT to $\mathbb{P}'_0$ satisfy the following properties:

$$\begin{aligned}
OPT(\Pi) = 1 &\implies OPT(\Pi') = C'_{t1} \\
OPT(\Pi) < \frac{1}{\epsilon} &\implies OPT(\Pi') > C'_{t1}(1+\frac{1}{120N}(1-\frac{1}{\epsilon}))
\end{aligned} \tag{17}$$

From [32], we know that there exists a polynomial-time reduction f_1 from SAT to MAX-3SAT such that, for some fixed $\epsilon > 1$, reduction f_1 satisfies

$$\begin{aligned}
I \in SAT &\implies \text{MAX-3SAT}(f_1(I)) = 1 \\
I \notin SAT &\implies \text{MAX-3SAT}(f_1(I)) < \frac{1}{\epsilon}
\end{aligned} \tag{18}$$

Then, Equation 17 and 18 imply the following:

$$\begin{aligned}
I \in SAT &\implies OPT(f(f_1(I))) = C'_{t1} \\
I \notin SAT &\implies OPT(f(f_1(I))) > C'_{t1}(1+\frac{1}{120N}(1-\frac{1}{\epsilon}))
\end{aligned} \tag{19}$$

Therefore, it is NP-hard to achieve an approximation ratio of $1 + \frac{1}{120N}(1 - \frac{1}{\epsilon})$ for problem $\mathbb{P}_0$.

We relegate the details to the appendix.

Based on the definition of polynomial time approximation scheme (PTAS) and Theorem 5, we then have

Corollary 2 *There is no polynomial time approximation scheme (PTAS) for problem* $\mathbb{P}_0$ *when* $K = 2$ *and re-aggregation is allowed.*

Based on the relations among $\mathbb{P}_0$, $\mathbb{P}_1$, $\mathbb{P}_2$, and $\mathbb{P}$, we have

Theorem 6 *When* $K = 2$ *and re-aggregation is allowed, problems* $\mathbb{P}_1$*,* $\mathbb{P}_2$*, and* $\mathbb{P}$ *are strong NP-hard whether or not the routing structure is a tree or a linear chain, and there is no polynomial-time approximation scheme (PTAS) for solving them.*

Proof The proof is similar to that of Theorem 3.

Theorems 4 and 6 show that, when $K = 2$ and re-aggregation is allowed, the joint optimization problems are strong NP-hard whether or not the routing structure is a tree or a linear chain, and whether or not the information elements are of the same length. That is, the complexity of these problems when $K = 2$ and re-aggregation is allowed is very much similar to the case when $K \geq 3$.

When re-aggregation is prohibited

When $K = 2$ and re-aggregation is prohibited, we can solve problem $\mathbb{P}$ (and thus its special versions $\mathbb{P}_0$, $\mathbb{P}_1$, and $\mathbb{P}_2$) in polynomial time by transforming it into a maximum weighted matching problem in an interval graph. An interval graph G_I is a graph defined on a set I of intervals on the real line such that 1) G_I has one and only one vertex for each interval in the set, and 2) there is an edge between two vertices if the corresponding intervals intersect with each other. Given an instance of problem $\mathbb{P}$, we solve it using Algorithm 1 as follows:

For Algorithm 1, we have

Theorem 7 *When* $K = 2$ *and re-aggregation is prohibited, Algorithm 1 solves problem* $\mathbb{P}$ *in* $O(n^3)$ *time, where* n *is the number of information elements considered in the problem.This*

Algorithm 1 Algorithm for solving $\mathbb{P}$ when $K = 2$ and re-aggregation is prohibited

1: Generate an interval graph $G_I(V_I, E_I)$ for problem $\mathbb{P}$ as follows:

- Select an arbitrary information element q generated by node v_q at time r_q and with spare time s_q, define an interval $[r_q, r_q + s_q]$ for q on the real line.
- For each remaining information element p generated by node v_p at time r_p and with spare time s_p, let node v_{pq} be the common ancestor of v_p and v_q that is the farthest away from R among all common ancestors of v_p and v_q, then define an interval $[r_q - t_{v_q v_{pq}} + t_{v_p v_{pq}}, r_q - t_{v_q v_{pq}} + t_{v_p v_{pq}} + s_q]$ for information element p.
- Let $V_I = \emptyset$. Then, for each information element s, define a vertex s and add it to V_I.
- Let $E_I = \emptyset$. If the two intervals that represent any two information elements u and h overlap with each other, define an edge (u, h) and add it to E_I; then assign edge (u, h) with a weight $com(u, h) = ETX_{v_{uh}R}(l_u) + ETX_{v_{uh}R}(l_h) - ETX_{v_{uh}R}(l_u + l_h)$, where l_u and l_h are the length of u and h respectively.

2: Solve the maximum weighted matching problem for G_I using Edmonds' Algorithm [24].

3: For each edge (u, h) in the matching, information elements u and h are packed together at node v_{uv}. For all other vertices not in the matching, their corresponding information elements are sent to the sink alone without being packed with any other information element.

holds whether or not the routing structure is a tree or a linear chain, and whether or not the information elements are of equal length.

Proof It is easy to see that if information elements u and h are packed together, the total number of transmissions taken to deliver u and h is $ETX_{v_u R}(l_u) + ETX_{v_h R}(l_h) - ETX_{v_{uh}R}(l_u) - ETX_{v_{uh}R}(l_h) + ETX_{v_{uh}R}(l_u + l_h) = ETX_{v_u R}(l_u) + ETX_{v_h R}(l_h) - com(u, h)$. Let V_I be the set of vertices in the interval graph G_I, M be a matching in G_I, V_1 be the set of nodes in M, and $V_2 = V_I / V_1$. Then the weight of M, denoted by W_M, is expressed in Equation 20:

Note that $\sum_{v \in V_I} ETX_{vR}(l_v)$ is a fixed value, and $\sum_{(u,h) \in M}[ETX_{v_u R}(l_u) + ETX_{v_h R}(l_h) - com(u, h)] + \sum_{v \in V_2} ETX_{vR}(l_v)$ is the total number of transmissions, denoted by ETX_{total}, incurred in the packing scheme generated by Algorithm 1. Therefore, ETX_{total} is minimized if and only if W_M is maximized, which means that solving the maximum weighted matching problem can give us an optimal solution to the original packet packing problem.

$$
\begin{aligned}
W_M &= \textstyle\sum_{(u,h)\in M} com(u,h) \\
&= \textstyle\sum_{(u,h)\in M}[ETX_{v_uR}(l_u) + ETX_{v_hR}(l_h) \\
&\quad -(ETX_{v_uR}(l_u) + ETX_{v_hR}(l_h) \\
&\quad -com(u,h))] \\
&= \textstyle\sum_{(u,h)\in M}(ETX_{v_uR}(l_u) + ETX_{v_hR}(l_h)) \\
&\quad -\textstyle\sum_{(u,h)\in M}[ETX_{v_uR}(l_u) + ETX_{v_hR}(l_h) \\
&\quad -com(u,h)] \\
&= \textstyle\sum_{s\in V_1} ETX_{sR}(l_s) + \sum_{v\in V_2} ETX_{vR}(l_v) \\
&\quad -\{\textstyle\sum_{(u,h)\in M}[ETX_{v_uR}(l_u) + ETX_{v_hR}(l_h) \\
&\quad -com(u,h)] \\
&\quad +\textstyle\sum_{v\in V_2} ETX_{vR}(l_v)\} \\
&= \textstyle\sum_{v\in V_I} ETX_{vR}(l_v) \\
&\quad -\{\textstyle\sum_{(u,h)\in M}[ETX_{v_uR}(l_u) + ETX_{v_hR}(l_h) \\
&\quad -com(u,h)] + \textstyle\sum_{v\in V_2} ETX_{vR}(l_v)\}
\end{aligned}
\tag{20}
$$

Let n denote the total number of information elements in this problem. The whole algorithm consists of three parts. The first one is to define an interval graph and assign weights to each node and edge in the graph, whose time complexity is $O(n^2)$. The second part is to solve the maximum weighted matching problem, whose time complexity is $O(n^3)$ by Edmonds' Algorithm [24]. And the third part is to convert the optimal matching problem to the optimal packing scheme, whose time complexity is $O(n)$. Therefore, the time complexity of the whole algorithm is $O(n^2) + O(n^3) + O(n) = O(n^3)$. ∎

By the definition of the weight $com(u,h)$ for elements u and h in Algorithm , the solution generated by the maximum weighted matching tends to greedily pack elements as soon as possible after they are generated. This observation motivates us to design a local, greedy online algorithm *tPack* in the next section for the general joint optimization problems, and the effectiveness of this approach will be demonstrated through competitive analysis and testbed-based measurement study in next two sections. Note that, incidentally, Theorem 7 also answers the open question on the complexity of scheduling batch-processes with release times in interval graphs [22].

A Utility-based online algorithm

We see from the complexity study section that problem $\mathbb{P}$ and its special cases in sensornets are strong NP-hard in most system settings, and there is no polynomial-time approximation scheme (PTAS) for these problems. Instead of trying to find global optimal solution, therefore, we focus on designing a distributed, approximation algorithm *tPack* that optimizes the local utility of packet packing at each node. Given that packet arrival processes are usually unknown a priori, we consider the online version of the optimization problem.

Based on the definition of $\mathbb{P}$, its optimization objective is to minimize

$$AC = \frac{TX_{net}}{\sum_{x \in X} l_x} \tag{21}$$

where TX_{net} is the total number of transmissions taken to deliver each information element $x \in X$ to the sink before its deadline. For convenience, we call AC the *amortized cost* of delivering $\sum_{x \in X} l_x$ amount of data. In what follows, we design an online algorithm tPack based on this concept of amortized cost of data transmission. Accordingly, a local optimization objective at a node j is to minimize

$$AC_j = \frac{TX_j}{data_j} \tag{22}$$

where TX_j is the total number of transmissions taken to deliver $data_j$ amount of data from j to the sink R before their deadline. Then an online algorithm, which we denote as *tPack*, is to minimize AC_j for the timely delivery of the data that node j currently holds.

When node j has a packet pkt in its data buffer, j can decide to transmit pkt immediately or to hold it. If j transmits pkt immediately, information elements carried in pkt may be packed with packets at j's ancestors to reduce the amortized cost of data transmissions from those nodes; if j holds pkt, more information elements may be packed with pkt so that the amortized cost of transmission from j can be reduced. Therefore, we can define the *utility* of transmitting or holding pkt as the expected reduction in amortized data transmission cost as a result of the corresponding action, and then the decision on whether to transmit or to hold pkt depends on the utilities of the two actions. For simplicity and for low control overhead, we only consider the immediate parent of node j when computing the utility of transmitting pkt. We will show the goodness of this local approach through competitive analysis later in this section and through testbed-based measurement in next section.

In what follows, we first derive the utilities of holding and transmitting a packet, then we present a scheduling rule that improves the overall utility.

Utility calculation

For convenience, we define the following notations:

L	:	maximum payload length per packet;
$ETX_{jp}(l)$	:	expected number of transmissions taken to transport a packet of length l from node j to its ancestor p;
p_j	:	the parent of node v_j in the routing tree.

The utilities of holding and transmitting a packet pkt at a node v_j depend on the following parameters related to traffic pattern:

- With respect to v_j itself and its children:

  r_l : expected rate in receiving another packet pkt' from a child or locally from an upper layer;

  s_l : expected payload size of pkt'.

- With respect to the parent of v_j:

r_p : expected rate for the parent to transmit another packet pkt'' that does not contain information elements generated or forwarded by v_j itself;

s_p : expected payload size of pkt''.

The utilities of holding and transmitting a packet pkt also depend on the following constraints posed by timeliness requirement for data delivery as well as limited packet size:

- Grace period t'_f for delivering pkt: the maximum allowable latency in delivering pkt minus the maximum time taken to transport pkt from v_j to the sink without being held at any intermediate node along the route.

  If $t'_f \leq 0$, pkt should be transmitted immediately to minimize the extra delivery latency.

- Spare packet space s'_f of pkt: the maximum allowable payload length per packet minus the current payload length of pkt.

  Parameter s'_f and the size of the packets coming next from an upper layer at v_j or from v_j's children determine how much pkt will be packed and thus the potential utility of locally holding pkt.

In the design and analysis of this section, we assume that packet arrival process (i.e.,, r_l, r_p), packet payload size and spare space (i.e., s_l, s_p, s'_f), and grace period (i.e., t'_f) are independent of one another. Then, the utilities of holding and transmitting a packet are calculated as follows.

Utility of holding a packet. When a node v_j holds a packet pkt, pkt can be packed with incoming packets from v_j's children or from an upper layer at v_j. Therefore, the utility of holding pkt at v_j is the expected reduction in the amortized cost of transmitting pkt after packing pkt. The utility depends on (a) the expected number of packets that v_j will receive within t'_f time (either from a child or locally from an upper layer), and (b) the expected payload size s_l of these packets. Given that the expected packet arrival rate is r_l, the expected number of packets to be received at v_j within t'_f time is $t'_f r_l$. Thus, the expected overall size $\mathcal{S}'_l$ of the payload to be received within t'_f time is

$$\mathcal{S}'_l = \frac{t'_f}{t_l} s_l$$

Given the spare space s'_f in the packet pkt, the expected size $\mathcal{S}_l$ of the payload that can be packed into pkt can be approximated[2] as

$$\mathcal{S}_l = \min\{\mathcal{S}'_l, s'_f\} = \min\{\frac{t'_f}{t_l} s_l, s'_f\}$$

Therefore, the expected amortized cost AC_l of transporting the packet to the sink R after the anticipated packing can be approximated as[2]

$$AC_l = \frac{1}{L - s'_f + \mathcal{S}_l} ETX_{jR}(L - s'_f + \mathcal{S}_l)$$

where $(L - s'_f)$ is the payload length of pkt before packing.

Since the amortized cost AC'_l of transporting pkt without the anticipated packing is

$$AC'_l = \frac{1}{L - s'_f} ETX_{jR}(L - s'_f)$$

the utility U_l of holding pkt is

$$U_l \quad = \quad AC'_l - AC_l \tag{23}$$

Utility of immediately transmitting a packet. If node v_j transmits the packet pkt immediately to its parent p_j, the utility comes from the expected reduction in the amortized cost of packet transmissions at p_j as a result of receiving the payload carried by pkt. When v_j transmits pkt to p_j, the grace period of pkt at p_j is still t'_f, thus the expected number of packets that do not contain information elements from v_j and can be packed with pkt at p_j is $t'_f r_p$, and we use P_{pkt} to denote this set of packets. Given the limited payload that pkt carries, it may happen that not every packet in P_{pkt} gets packed (to full) via the payload from pkt. Accordingly, the utility U_p of immediately transmitting pkt is calculated as follows:

- If *every packet in P_{pkt} gets packed to full* with payload from pkt, i.e., $t'_f r_p(L - s_p) \leq L - s'_f$:

[2]We use this approximation because it is usually difficult to estimate and store the complete distributions of random variables in resource-constrained sensor nodes.

Then, the overall utility U'_p can be approximated as [2]

$$U'_p = \frac{\frac{t'_f}{t_p} ETX_{p_j R}(s_p)}{\frac{t'_f}{t_p} s_p} - \frac{\frac{t'_f}{t_p} ETX_{p_j R}(L)}{\frac{t'_f}{t_p} L} = \frac{ETX_{p_j R}(s_p)}{s_p} - \frac{ETX_{p_j R}(L)}{L} \tag{24}$$

- If *not every packet in P_{pkt} gets packed to full* with payload from pkt, i.e., $t'_f r_p(L - s_p) > L - s'_f$:

  In this case, $\lfloor\frac{L-s'_f}{L-s_p}\rfloor$ number of packets are packed to full; if $\text{mod}(L - s'_f, L - s_p) > 0$, there is also a packet that gets partially packed with $\text{mod}(L-s'_f, L-s_p)$ length of payload from pkt. Thus the total number of packets that benefit from the packet transmission is $\lceil\frac{L-s'_f}{L-s_p}\rceil$. Denoting $\text{mod}(L - s'_f, L - s_p)$ by l_{mod} and letting I_{mod} be 1 if $l_{mod} > 0$ and 0 otherwise, then the overall utility U''_p can be approximated as[2]

$$U''_p = \frac{\lceil\frac{L-s'_f}{L-s_p}\rceil ETX_{p_j R}(s_p)}{\lceil\frac{L-s'_f}{L-s_p}\rceil s_p} - \frac{\lfloor\frac{L-s'_f}{L-s_p}\rfloor ETX_{p_j R}(L) + I_{mod} ETX_{p_j R}(s_p + l_{mod})}{\lceil\frac{L-s'_f}{L-s_p}\rceil s_p + L - s'_f} \tag{25}$$

Therefore, the utility U_p of immediately transmitting pkt to p_j can be computed as

$$U_p = \begin{cases} U'_p & \text{if } t'_f r_p(L - s_p) \leq L - s'_f \\ U''_p & \text{otherwise} \end{cases} \tag{26}$$

where U'_p and U''_p are defined in Equations (24) and (25) respectively.

Scheduling rule

Given a packet to be scheduled for transmission, if the probability that the packet is immediately transmitted is P_t ($0 \leq P_t \leq 1$), then the expected utility $U_t(P_t)$ is

$$\begin{aligned} U_t(P_t) &= P_t \times U_p + (1 - P_t)U_l \\ &= U_l + P_t(U_p - U_l) \end{aligned} \tag{27}$$

where U_p and U_l are the utilities of immediately transmitting and locally holding the packet respectively. To maximize U_t, P_t should be set according to the following rule:

$$P_t = \begin{cases} 1 & \text{if } U_p > U_l \\ 0 & \text{otherwise} \end{cases}$$

That is, the packet should be immediately transmitted if the utility of immediate transmission is greater than that of locally holding the packet. For convenience, we call this local, distributed decision rule *tPack* (for *time-sensitive packing*). Interested readers can find the discussion on how to implement tPack in TinyOS in [68].

Competitive analysis

To understand the performance of tPack as compared with an optimal online algorithm, we analyze the competitive ratio of tPack. Since it is difficult to analyze the competitive ratio of non-oblivious online algorithms for arbitrary network and traffic pattern in the joint optimization and tPack is a non-oblivious algorithm, we only study the competitive ratio of tPack for complete binary trees where all the leaf nodes generate information elements according to a common data generation process, and we do not consider the impact of packet length on link ETX. We denote these special cases of problem $\mathbb{P}$ as problem $\mathbb{P}'$. The theoretical analysis here is to get an intuitive understanding of the performance of tPack; we experimentally analyze the behaviors of tPack with different networks, traffic patterns, and application requirements through testbed-based measurement in the performance evaluation section. We relegate the study on the competitive ratio of tPack as well as the lower bound on the competitive ratio of non-oblivious online algorithms for the general problem $\mathbb{P}$ as a part of our future work. (Note that the best results so far on the lower bound of the competitive ratio of joint INP- and latency-optimization also only considered the cases where only leaf nodes generate information elements [59], and these results are for oblivious algorithms and for cases where no aggregation constraint is considered [59].)

Theorem 8 *For problem $\mathbb{P}'$, tPack is $\min\{K, \max_{v_j \in V_{>1}} \frac{2ETX_{v_j R}}{2ETX_{v_j R} - ETX_{p_j R}}\}$-competitive, where K is the maximum number of information elements that can be packed into a single packet, $V_{>1}$ is the set of nodes that are at least two hops away from the sink R.*

Proof For convenience, we denote the optimal packing scheme as OPT. By definition, tPack is at least K-competitive since, considering the packets transmitted by a given node v_i in the routing tree, the length of the packet containing an information element x in OPT is no more than K times the length of the packet containing x in tPack.

To get a tighter performance bound for tPack, we first analyze the packet length for the packets transmitted by a leaf node v_j. Suppose that v_j transmits a packet pkt with length l_{pkt} when the latency requirement could have allowed packing another l' amount of data with the packet. In this case, the utility of holding pkt is

$$U_l = \frac{ETX_{v_j R}}{l_{pkt}} - \frac{ETX_{v_j R}}{l_{pkt} + l'} = ETX_{v_j R} \frac{l'}{l_{pkt}(l_{pkt} + l')} \tag{28}$$

By definition, the utility of immediately transmitting pkt is no more than the transmission utility that would be generated if the information elements of pkt are all packed into another packet pkt^* at p_j, the parent of v_j, that was transmitted to p_j from its the child other than v_j. Given that the routing tree is a complete binary tree and that the leaf nodes generate information elements according to a common data generation process, the lengths of packets that are transmitted along links at the same tree level are expected to be the same. Thus we can assume that the payload length of pkt^* is also l_{pkt}. Therefore, the utility of immediately forwarding pkt

at v_j satisfy the following inequality

$$U_p \leq \frac{ETX_{p_jR}}{l_{pkt}} - \frac{ETX_{p_jR}}{l_{pkt} + l_{pkt}} = \frac{ETX_{p_jR}}{2l_{pkt}} \tag{29}$$

By the design of tPack, we know that $U_l < U_p$. From (28) and (29), thus we have

$$ETX_{v_jR}\frac{l'}{l_{pkt}(l_{pkt} + l')} < \frac{ETX_{p_jR}}{2l_{pkt}}$$

Thus

$$l' < \frac{a}{2-a}l_{pkt} \tag{30}$$

where $a = \frac{ETX_{p_jR}}{ETX_{v_jR}}$.

Due to the constraint imposed by application's requirement on the timeliness of data delivery, we know that the length of the packet, denoted by l_{opt}, that contains the information elements of pkt in OPT is no more than $l_{pkt} + l'$. Then from (30), we know that

$$l_{opt} \leq l_{pkt} + l' < \frac{2}{2-a}l_{pkt} = \frac{2ETX_{v_jR}}{2ETX_{v_jR} - ETX_{p_jR}}l_{pkt}$$

That is,

$$\frac{l_{opt}}{l_{pkt}} < \frac{2ETX_{v_jR}}{2ETX_{v_jR} - ETX_{p_jR}} \tag{31}$$

For a node v_i that is not a leaf node, the same analysis applies. Given a packet pkt' of length $l_{pkt'}$ that is transmitted by v_i when the latency requirement could have allowed packing another l'' amount of data with pkt', we have

$$l'' < \frac{a'}{2-a'}l_{pkt'} \tag{32}$$

where $a' = \frac{ETX_{p_iR}}{ETX_{v_iR}}$. Moreover, the length of the packet, denoted by $l_{opt'}$, that contains the information elements of pkt' in OPT is no more than $l_{pkt'} + l''$; this is due to the following reasons:

- If a packet pkt_{max} contains $l_{pkt'} + l''$ amount of data payload without constrained by packet size limit, then the spare time of pkt_{max} is 0.

- Consider a packet pkt'' transmitted by v_i in OPT whose length is $l_{opt'}$. If v_i holds pkt'' until its spare time is 0 (instead of transmitting pkt'') in OPT, the resulting length of the new packet pkt''_0 is no more than $l_{pkt'} + l''$. This is because data flows faster toward the sink in tPack as compared with OPT, and pkt' reaches v_i earlier than pkt'' does.

- Therefore, $l_{opt'}$ is no more than the length of pkt''_0, which is no more than $l_{pkt'} + l''$. Thus, $l_{opt'} \leq l_{pkt'} + l''$

Therefore, we have

$$\frac{l_{opt'}}{l_{pkt'}} < \frac{2ETX_{v_iR}}{2ETX_{v_iR} - ETX_{p_iR}} \tag{33}$$

From (31) and (33), we know that tPack is at least $O(\max_{v_j \in V_{>1}} \frac{2ETX_{v_jR}}{2ETX_{v_jR} - ETX_{p_jR}})$-competitive. Therefore, tPack is $\min\{K, \max_{v_j \in V_{>1}} \frac{2ETX_{v_jR}}{2ETX_{v_jR} - ETX_{p_jR}}\}$-competitive for problem $\mathbb{P}'$.

From Theorem 8, we see that tPack is 2-competitive if every link in the network is of equal ETX value.

Implementation

From the discussion in last section, a node v_j needs to obtain the following parameters when calculating the utilities of holding and transmitting a packet:

- On routing tree: $ETX_{jR}(l)$, p_j, and $ETX_{p_jR}(l)$;

- On traffic pattern: r_l, s_l, r_p, s_p, and K.

Parameters related to routing tree can be provided by the routing component in a given system platform. Given a link $\langle j, p \rangle$, $ETX_{jp}(l)$ as a function of packet length l can be estimated using $ETX_{jp}(1)$, the ETX value of transmitting a packet of one unit length, as follows:

$$ETX_{jp}(l) = 1/(\frac{1}{ETX_{jp}(1)})^l = ETX_{jp}(1)^l$$

Accordingly, the routing component only needs to estimate $ETX_{jp}(1)$ instead of the ETX values for packets of arbitrary length.

For parameters related to traffic pattern, v_j can estimate by itself the parameters r_l and s_l, and K is readily available and fixed for each specific platform. To enable each node v_j to obtain parameters r_p and s_p, every node i in the network estimates the expected rate r_i to transmit two consecutive packets at i itself and the expected size s_i of these packets. Then, every node i shares with its neighbors the parameters r_i and s_i by piggybacking these information onto data packets or other control packets in the network. When a node v_j overhears parameter r_{p_j} and s_{p_j} from its parent p_j, v_j can approximate r_p and s_p with $r_{p_j} - r_j \frac{s_j}{s_{p_j}}$ and s_{p_j} respectively. The derivation is as follows.

Approximation of r_p *and* s_p: Since information elements generated or forwarded by the children of node p_j are treated in the same manner (without considering where they are from), the expected size of the packet being transmitted by p_j does not depend on whether the packet contains information elements generated or forwarded by v_j. Thus, v_j can simply regard s_{p_j} as s_p, the expected size of the packet transmitted by p_j that does not contain information elements coming from v_j.

Now we derive r_p as follows. Since the amount of payload transmitted by p_j per unit time is $r_{p_j} s_{p_j}$ and the amount of payload transmitted by v_j is $r_j s_j$ per unit time, the amount of payload l_p that are transmitted by p_j but are not from v_j per unit time is calculated as: $l_p = r_{p_j} s_{p_j} - r_j s_j$. Thus, the expected rate r_p that p_j transmits packets that do not contain information elements from v_j is calculated as: $r_p = l_p / s_{p_j} = r_{p_j} - r_j \frac{s_j}{s_{p_j}}$. Therefore, the expected interval t_p between p_j transmitting two consecutive packets that do not contain information elements from v_j is as follows: $t_p = \frac{1}{r_p} = \frac{t_{p_j} \times t_j \times s_{p_j}}{t_j \times s_{p_j} - t_{p_j} \times s_j}$.

Performance evaluation

To characterize the impact of packet packing and its joint optimization with data delivery timeliness, we experimentally evaluate the performance of tPack in this section. We first present the experimentation methodology and then the measurement results.

Methodology

Testbed. We use the *NetEye* wireless sensor network testbed at Wayne State University [3]. NetEye is deployed in an indoor office as shown in Figure 11. We use a 10×13 grid

Figure 11: *NetEye* wireless sensor network testbed

of TelosB motes in NetEye, where every two closest neighboring motes are separated by 2 feet. Out of the 130 motes in NetEye, we randomly select 120 motes (with each mote being selected with equal probability) to form a random network for our experimentation. Each of these TelosB motes is equipped with a 3dB signal attenuator and a 2.45GHz monopole antenna.

In our measurement study, we set the radio transmission power to be -25dBm (i.e., power level 3 in TinyOS) such that multihop networks can be created. We also use channel 26 of the CC2420 radio to avoid external interference from sources such as the campus WLANs. We use the TinyOS collection-tree-protocol (CTP) [27] as the routing protocol to form the routing structure, and we use the Iowa's Timesync protocol [2] for network wide time synchronization.

Protocols studied. To understand the impact of packet packing and its joint optimization with data delivery timeliness, we comparatively study the following protocols:[3]

- *noPack*: information elements are delivered without being packed in the network.
- *simplePack*: information elements are packed if they happen to be buffered in the same queue, but there is not packing-oriented scheduling.
- *SL*: the *spread latency* algorithm proposed in [10], where the spare time of an information element is evenly spent at each hop from its source to the sink without considering

[3] We use the terms protocols, algorithms, and decision rules interchangeably in this paper.

specific network conditions (e.g., network-wide traffic pattern). SL was proposed with total aggregation in mind without considering aggregation constraints such as maximum packet size.

- *CC*: the *common clock* algorithm proposed in [10], where the spare time of an information element is only partly spent at the node where it is generated. Same as SL, CC was proposed with total aggregation in mind.

- *tPack*: the packing- and timeliness-oriented scheduling algorithm that maximizes the local utility at each node, as we proposed in this chapter. (We have also evaluated another version of tPack, denoted by *tPack-2hop*, where the forwarding utility U_p considers both the parent node and the parent's parent; we find that tPack-2hop does not bring significant improvement over tPack while introducing higher overhead and complexity, thus our discussion here only focuses on tPack.)

We have implemented, in TinyOS [5], a system library which includes all the above protocols. The implementation takes 40 bytes of RAM (plus the memory required for regular packet buffers) and 4,814 bytes of ROM.

Performance metrics. For each protocol we study, we evaluate their behavior based on the following metrics:

- *Packing ratio*: number of information elements carried in a packet;

- *Delivery reliability*: percentage of information elements correctly received by the sink;

- *Delivery cost*: number of transmissions required for delivering an information element from its source to the sink;

- *Deadline catching ratio*: out of all the information elements received by the sink, the percentage of them that are received before their deadlines;

- *Latency jitter*: variability of the time taken to deliver information elements from the same source node, measured by the coefficient-of-variation (COV) [36] of information delivery latency.

Traffic pattern. To experiment with different sensornet scenarios, we use both periodic data collection traffic and event detection traffic trace as follows:

- $D3$: each source node periodically generates 50 information elements with an inter-element interval, denoted by Δ_r, uniformly distributed between 500ms and 3s; this is to represent high traffic load scenarios.
- $D6$: same as $D3$ except that Δ_r is uniformly distributed between 500ms and 6s; this is to represent relatively low traffic load scenarios.
- $D9$: same as $D3$ except that Δ_r is uniformly distributed between 500ms and 9s.
- E_{lites}: an event traffic where a source node generates one packet based on the Lites [1] sensornet event traffic trace.

To understand the impact of the timeliness requirement of data delivery, we experiment with different latency requirements. For periodic traffic, we consider maximum allowable latency in delivering information elements that is 1, 3, and 5 times the average element generation period, and we denote them by $L1$, $L3$, and $L5$ respectively; for event traffic, we consider maximum allowable latency that is 2s, 4s, or 6s, and we denote them by $L2'$, $L4'$, and $L6'$ respectively. Out of the 120 motes selected for experimentation, we let the mote closest to a corner of NetEye be the sink node, and the other mote serves as a traffic source if its node ID is even. For convenience, we regard a specific combination of source traffic model and latency requirement a *traffic pattern*. Thus we have 8 traffic patterns in total. To gain statistical insight, we repeat each traffic pattern 20 times. Note that, in each traffic pattern, all the information elements have the same maximum allowable latency. In our implementation, each information element is 16-byte long, and the TelosB motes allow for aggregating up to 7 information elements into a single packet (i.e., $K = 7$).

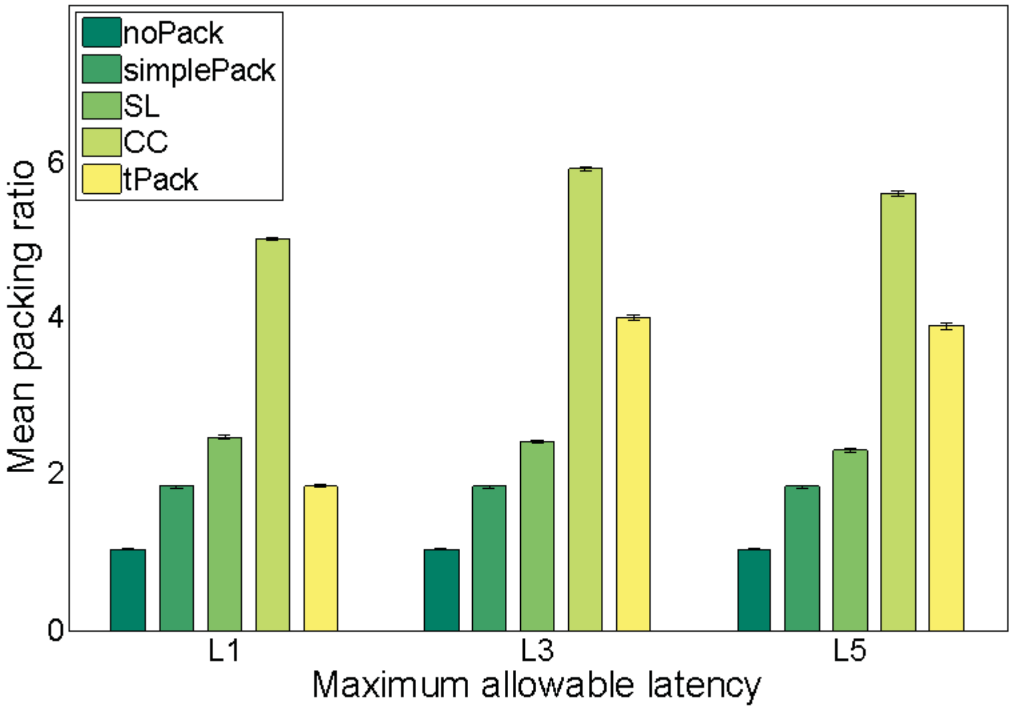

Figure 12: Packing ratio: $D3$

Measurement Results

In what follows, we first present the measurement results for periodic traffic patterns $D3$, $D6$, and $D9$, then we discuss the case of event traffic pattern E_{lites}. In most figures of this section, we present the means/medians and their 95% confidence intervals for the corresponding metrics such as the packing ratio, delivery reliability, delivery cost, deadline catching ratio, and the latency jitter.[4]

Periodic Data Traffic

For the periodic traffic pattern $D3$, Figures 12-16 show the packing ratio, delivery reliability, delivery cost, deadline catching ratio, and latency jitter in different protocols. tPack tends to enable higher degree of packet packing (i.e., larger packing ratio) than other protocols except the CC protocol. The increased packing in $tPack$ reduces channel contention and thus reduces the probability of packet transmission collision, which improves data delivery reliabil-

[4]The distributions for delivery reliability and latency jitter are not symmetric, thus we use medians instead of means to summarize their properties [36].

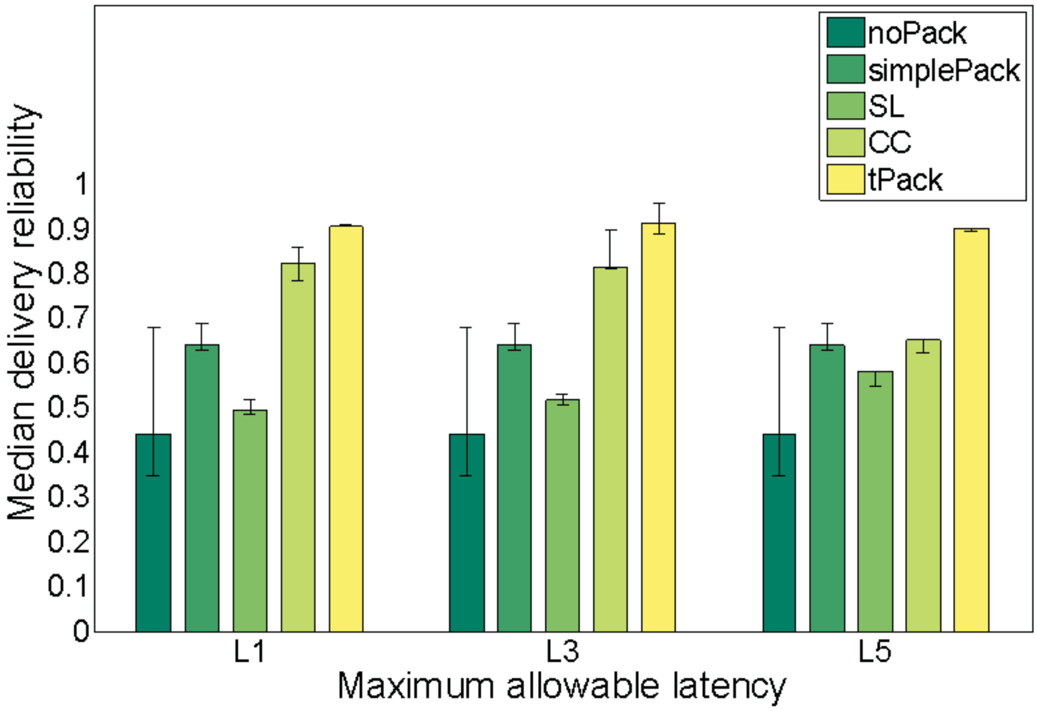

Figure 13: Delivery reliability: $D3$

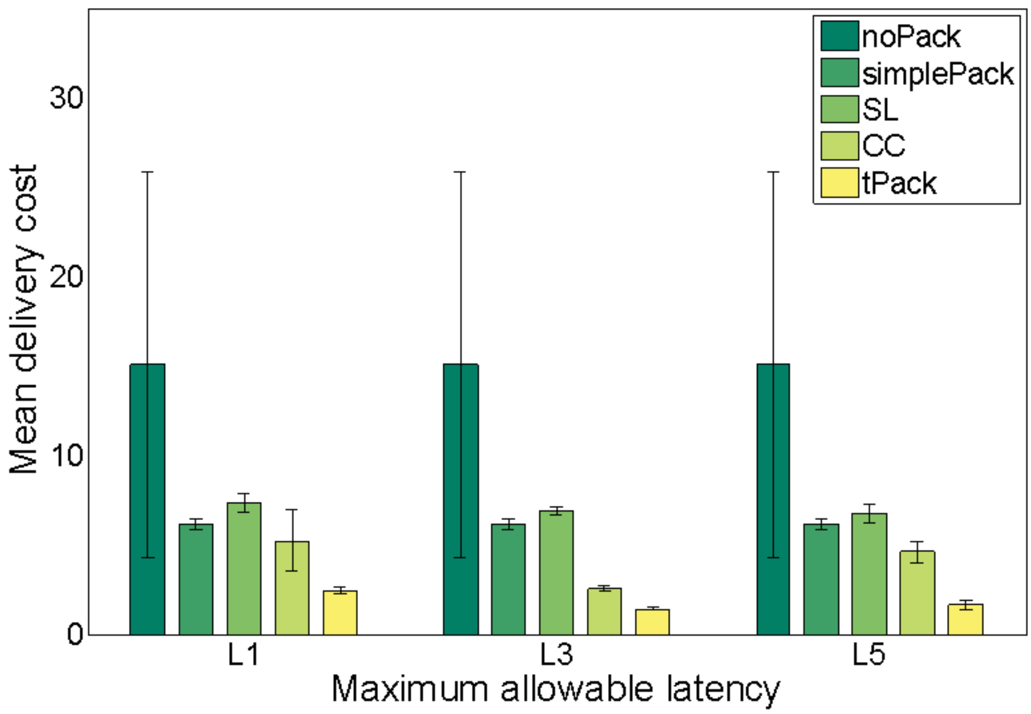

Figure 14: Delivery cost: $D3$

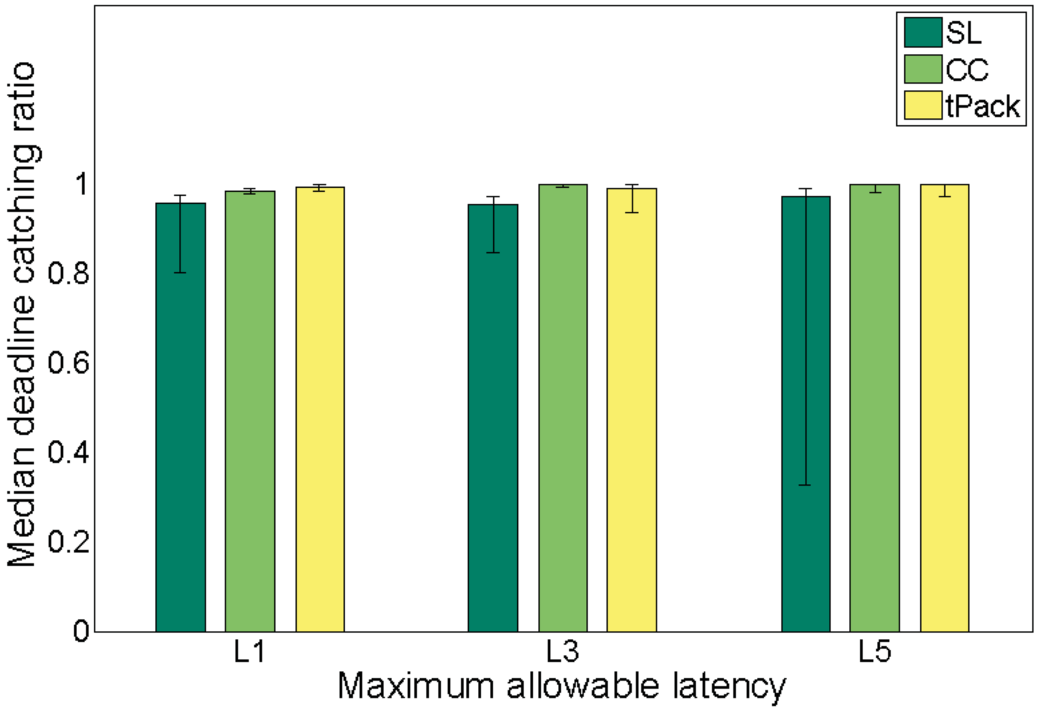

Figure 15: Deadline catching ratio: $D3$

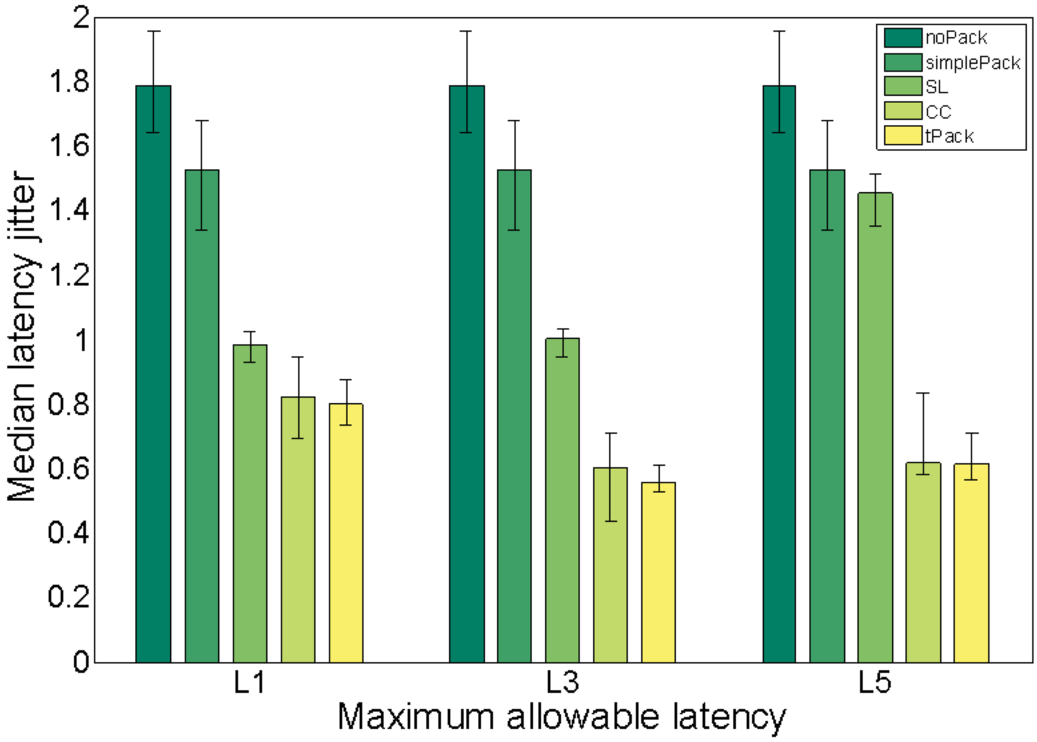

Figure 16: Latency jitter: $D3$

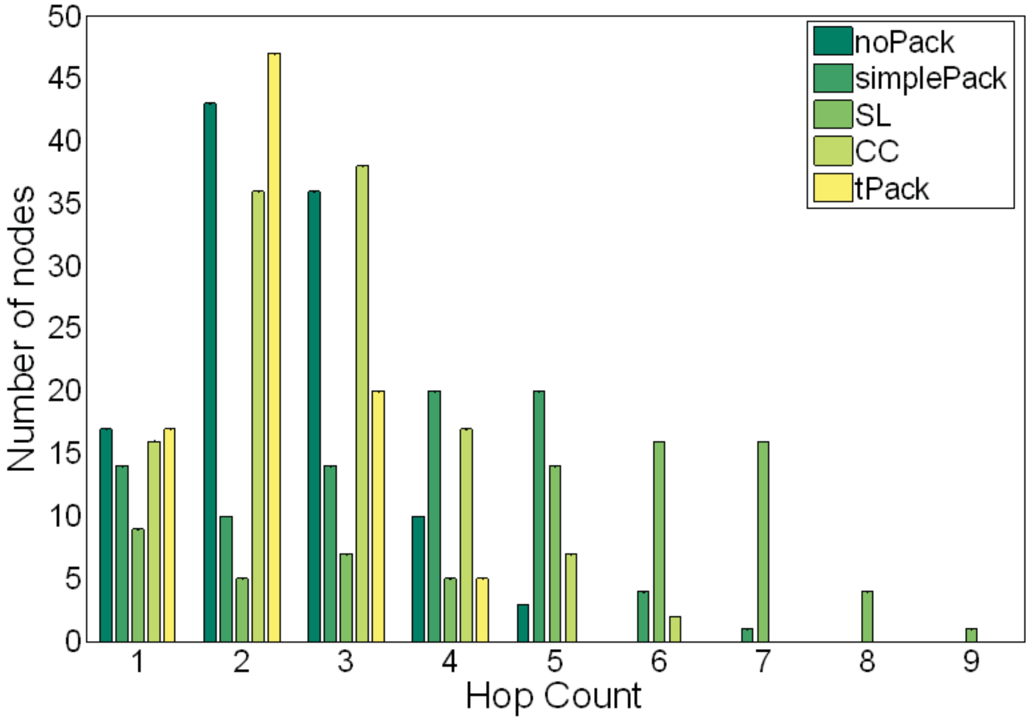

Figure 17: Histogram of routing hop count: $D3$ with maximum latency $L1$

ity. The reduced probability of transmission collision and the increased number of information elements carried per packet in $tPack$ in turn reduces delivery cost, since there are fewer number of packet retransmissions as well as fewer number of packets generated. Note that the low delivery reliability in simplePack is due to intense channel contention.

Exceptions to the above general observation happen in the case of maximum allowable latency $L1$ or when comparing tPack with CC. In the first case, the packing ratio in tPack is lower than that in SL, but tPack still achieves much higher delivery reliability (i.e., by more than 40%) and much lower delivery cost (i.e., by a factor of more than 3). This is because the packing ratio in SL is too high such that, in the presence of high wireless channel contention due to the high traffic load of $D3$ and the stringent real-time requirement of $L1$, the resulting long packet length leads to higher packet error rate and lower packet delivery reliability (as shown in Figure 13). The routing protocol CTP adapts to the higher packet error rate in SL, and this leads to longer routes and larger routing hops in SL. This can be seen from Figure 17 which shows the histogram of routing hop counts in different protocols. The maximum hop count in tPack is 4, whereas the hop count can be up to 9 in SL. Together, the higher packet error rate and the longer routes in SL lead to larger delivery cost in SL as compared with tPack. Similar arguments apply to the case when comparing tPack with CC. From these data on the benefits of tPack in comparison with SL and CC, we can see the importance of adapting to network conditions and data aggregation constraints in in-network processing. Note that similar arguments also explain the phenomenon where SL has higher packing ratio than simplePack but lower delivery reliability and higher delivery cost under all latency settings of $D3$ traffic.

Figure 13 also shows that tPack improves data delivery reliability even when the allowable latency in data delivery is small (e..g, in the case of $L1$) where the inherent probability for packets to be packed tends to be small. Therefore, tPack can be used for real-time applications where high data delivery reliability is desirable. Figure 12 shows that the packing ratio in tPack is close to 4 except for the case of $L1$ where 1) too much packing is undesirable as discussed earlier and 2) the packing probability is significantly reduced by the limited probability for a node to wait due to stringent timeliness requirement. Our offline analysis shows that the optimal packing ratio is $\sim$5 for the traffic patterns $D3$-$L3$ and $D3$-$L5$; thus tPack achieves a packing ratio very close to the optimal, which corroborates our analytical result in Theorem 8.

Figure 15 shows the deadline catching ratio in deadline-aware data aggregation schemes tPack, SL, and CC. Though the deadline catching ratio of all the three protocols are close to 1, the catching ratio of tPack is the highest and is greater than 0.99 in all cases. The slightly higher deadline catching ratio in tPack is a result of its online adaptation of packet holding time at each hop according to in-situ channel and traffic conditions along the path. As a result of the properly controlled packet packing, the reduced channel contention and improved packet delivery reliability in tPack also help enable lower performance variability. For instance, Figure 16 shows the latency jitter in different protocols, and we see that the jitter tends to be the lowest in tPack, especially when the real-time requirement is stringent (e.g., in $L1$ and $L3$). These properties are desirable in cyber-physical-system (CPS) sensornets where real-time sensing and control require predictable data delivery performance (e.g., in terms of low latency jitter), especially in the presence of potentially unpredictable, transient perturbations.

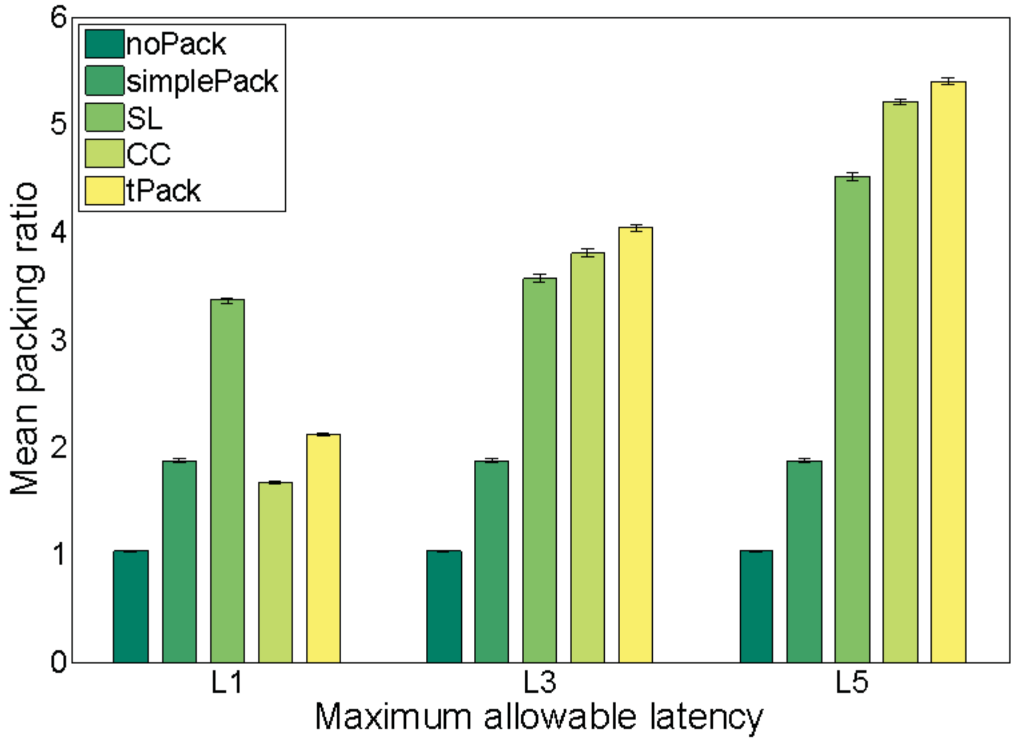

Figure 18: Packing ratio: $D6$

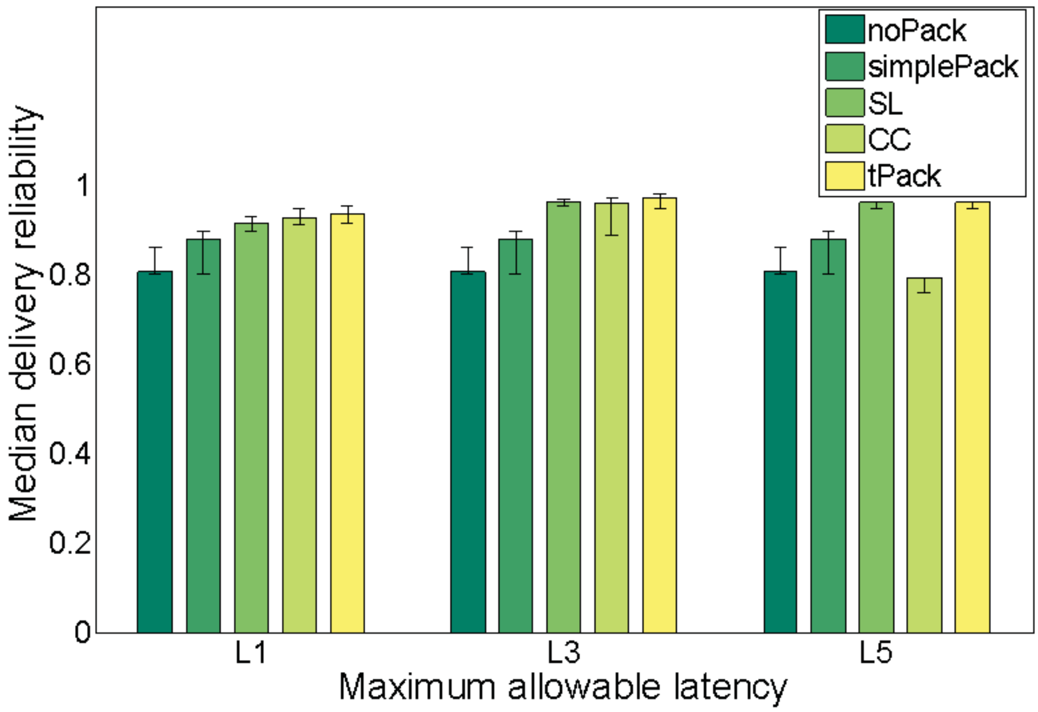

Figure 19: Delivery reliability: $D6$

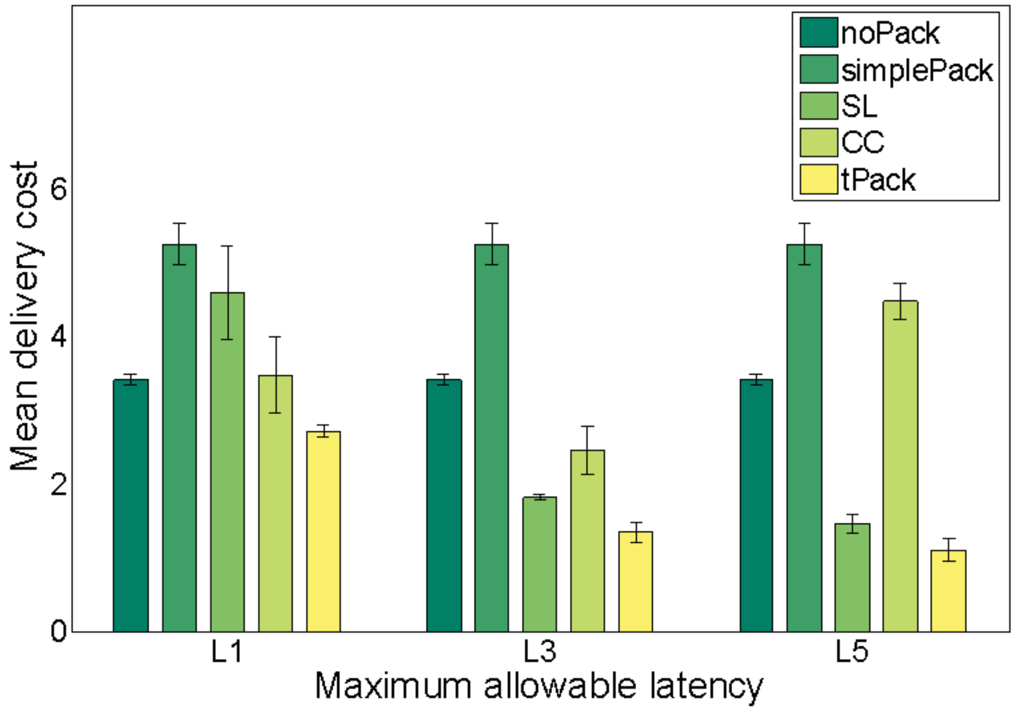

Figure 20: Delivery cost: $D6$

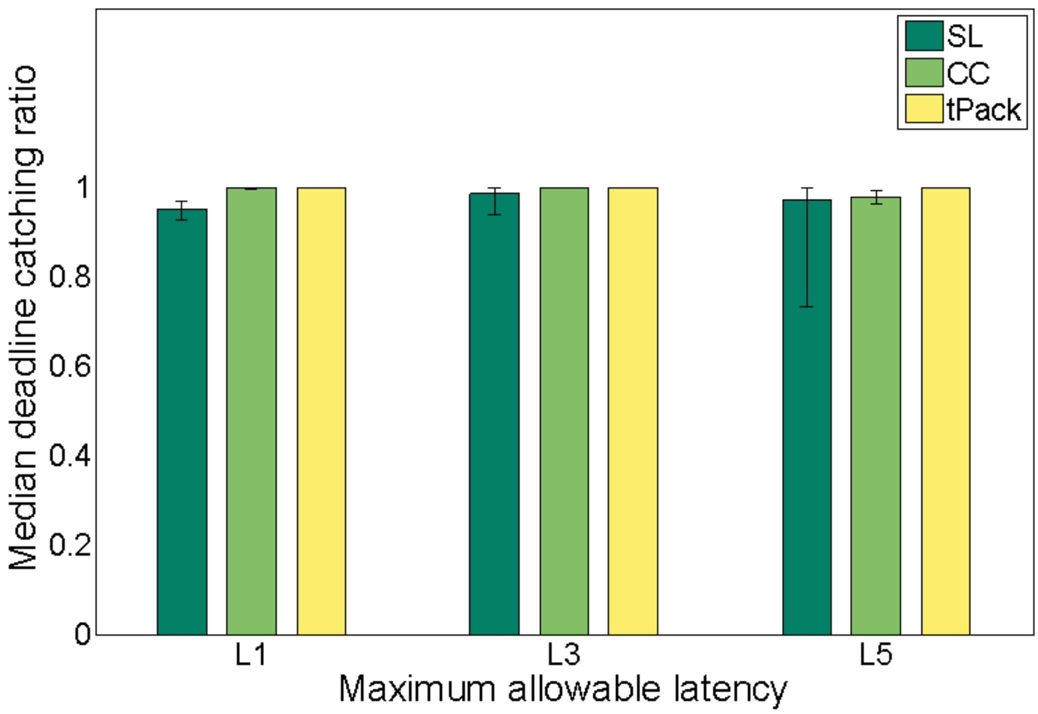

Figure 21: Deadline catching ratio: $D6$

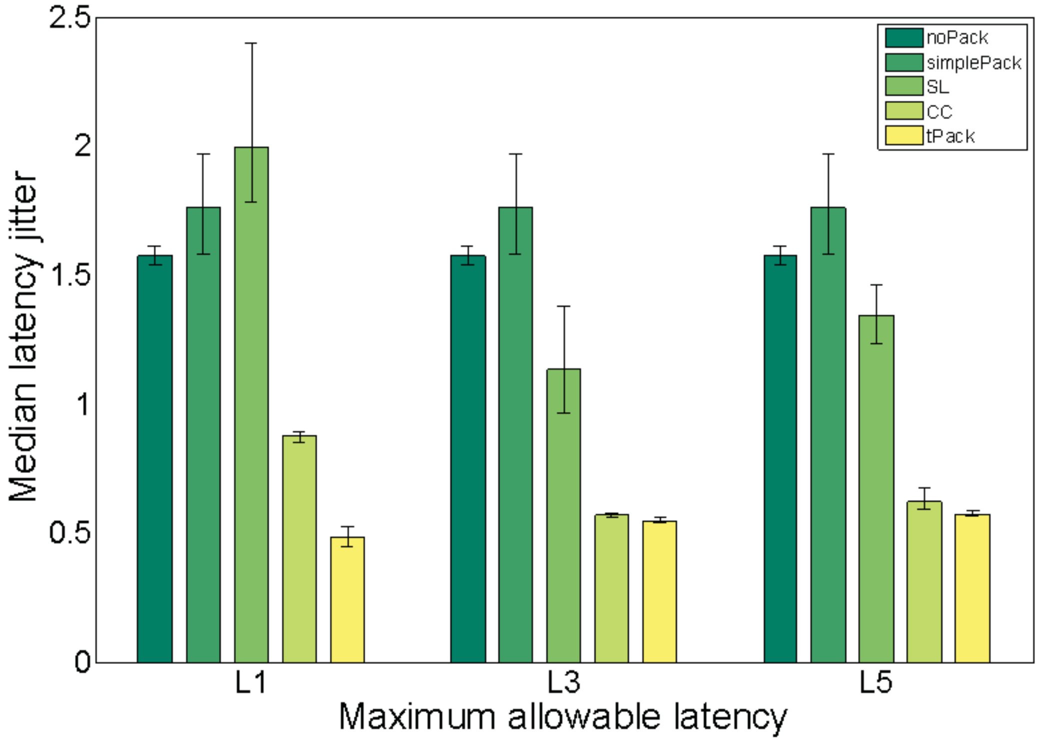

Figure 22: Latency jitter: $D6$

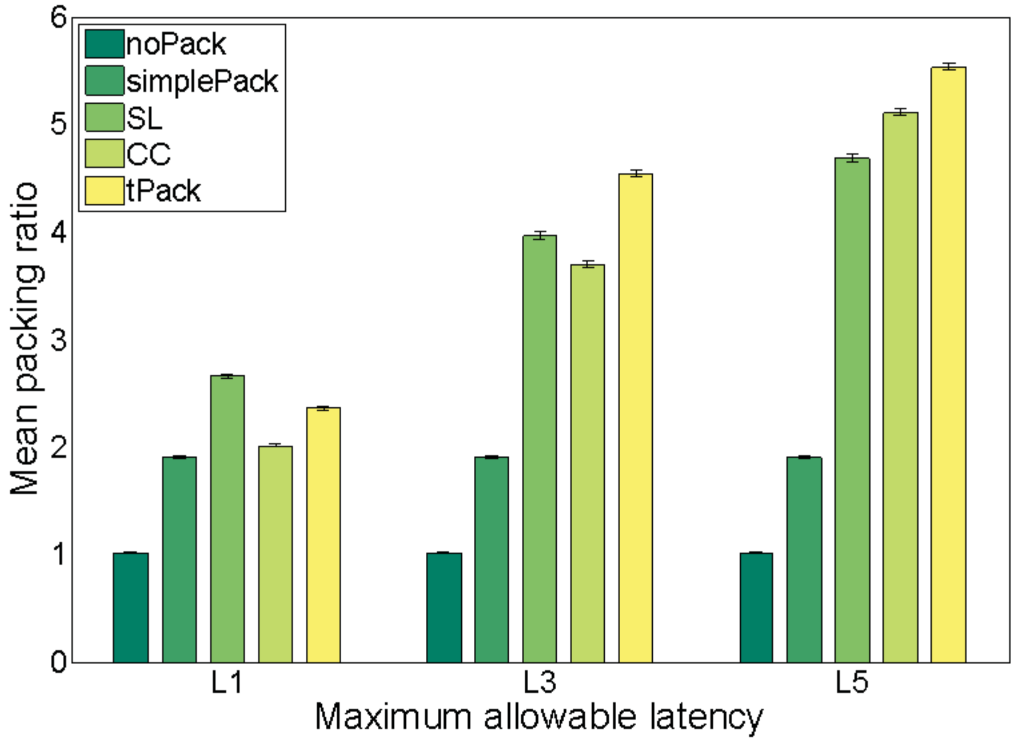

Figure 23: Packing ratio: $D9$

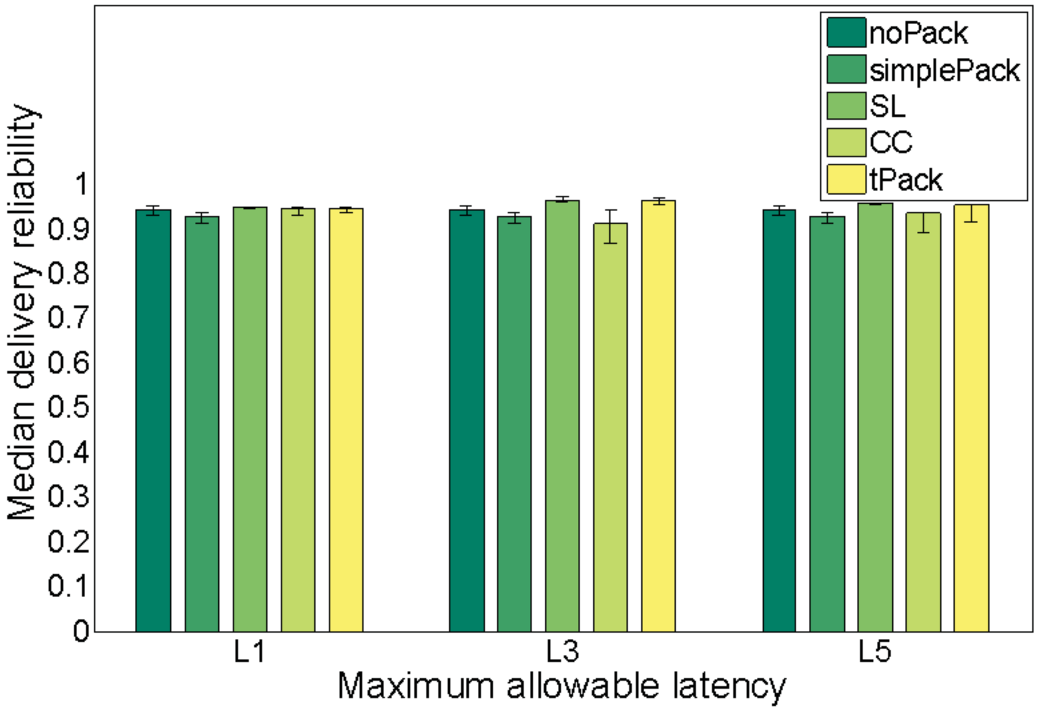

Figure 24: Delivery reliability: $D9$

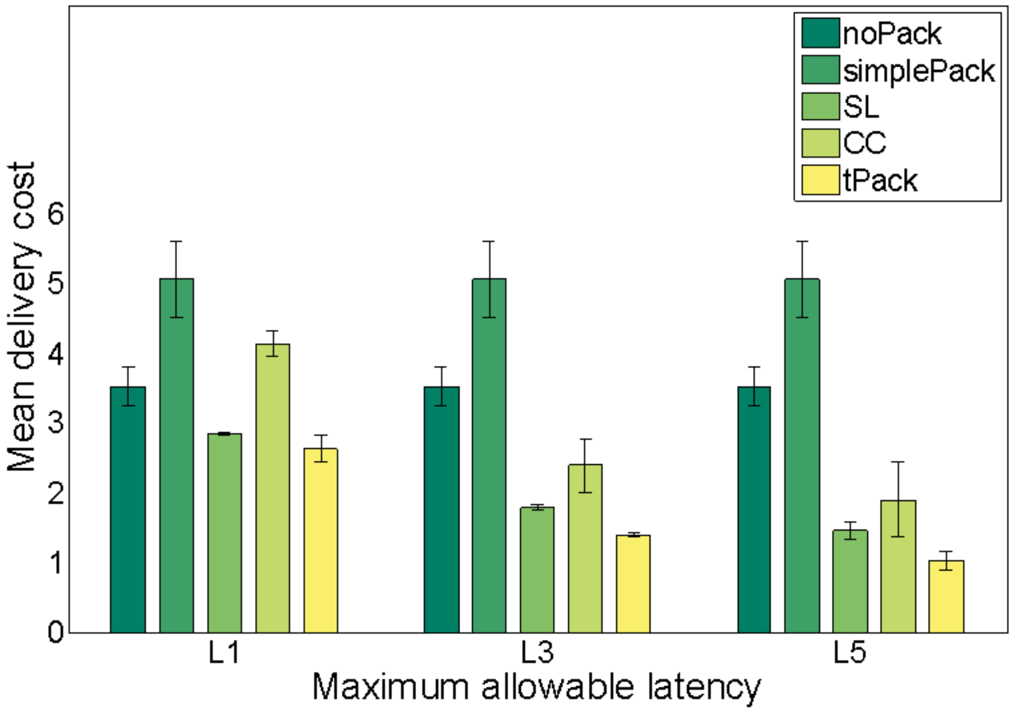

Figure 25: Delivery cost: $D9$

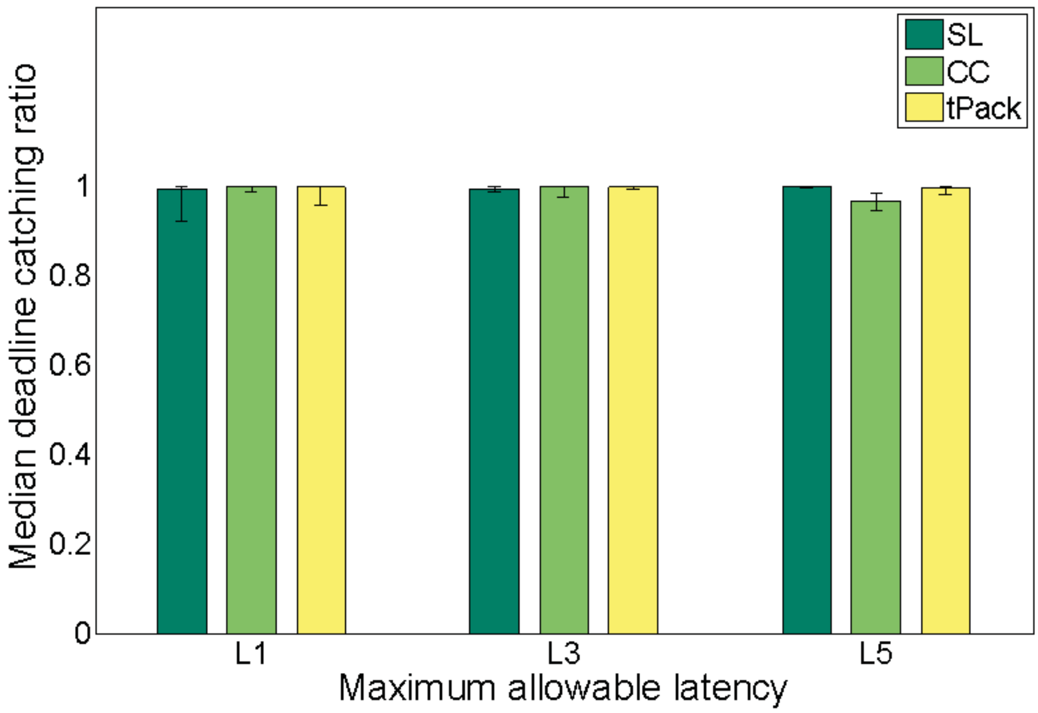

Figure 26: Deadline catching ratio: $D9$

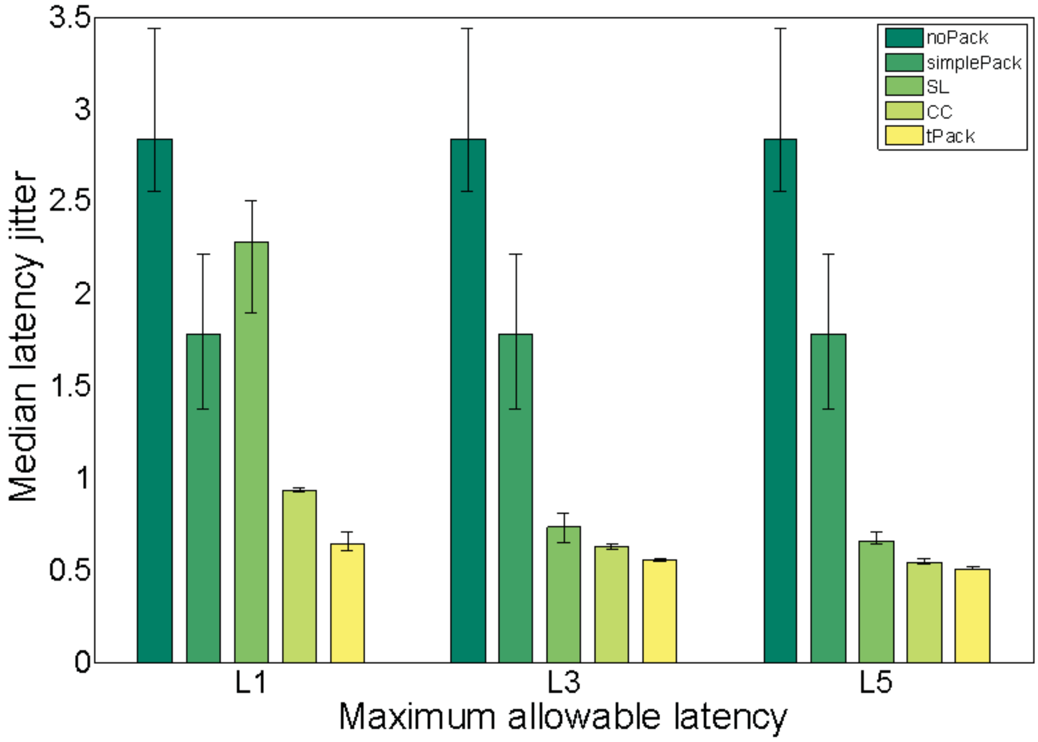

Figure 27: Latency jitter: $D9$

Figures 18-22 and Figures 23-27 show the measurement results for periodic traffic patterns $D6$ and $D9$ respectively. We see that, in terms of relative protocol performance, the overall trends in $D6$ and $D9$ are similar to those in $D3$. For instance, with stringent real-time requirement in $L1$, SL achieves a lower delivery reliability and a higher delivery cost than tPack even though the packing ratio tends to be higher in SL. Due to the reduced traffic load and thus the reduced wireless channel contention and collision, however, the delivery reliability of noPack, simplePack, and SL is also relatively high compared with their delivery reliability in $D3$.

Note that, in [10], CC is shown to have a much higher competitive ratio than SL through theoretical analysis. From our measurement study, however, we see that the performance of CC is not always better than SL. For instance, CC has a lower delivery reliability and a higher delivery cost than SL in $D6 - L5$. This seemingly discrepancy is due to the fact that the theoretical analysis of [10] does not consider the limit of data aggregation capacity, nor does it consider wireless link unreliability and interference in scheduling.

Surprisingly, Figures 18-20 show that, for the traffic pattern $D6$, simplePack introduces higher delivery cost than noPack does even though the packing ratio and the end-to-end delivery reliability are higher in simplePack. One reason for this is that, partially due to the increased packet length in simplePack, the link reliability in simplePack is lower than that in noPack as shown in Figure 28.[5] The routing protocol CTP adapts to the lower link reliability in simplePack and introduces longer routing hop length, which can be seen from Figure 29 which shows the histogram of routing hop counts for noPack and simplePack in traffic pattern $D6$-$L1$. Together, the lower link reliability and the longer routes in simplePack introduce larger information delivery cost when compared with noPack in $D6$. This observation is also corroborated by the detailed analysis of the cost (e.g., mean number of transmissions) taken to deliver an information element. For instance, Figure 30 shows the mean cost of delivering an information element from a node at different geographic distances (in terms of the number of grid hops) from the base station for the traffic pattern $D6$-$L1$. (Similar phenomena are observed for other

[5] The reason why simplePack still has higher end-to-end information element delivery reliability despite its lower link reliability is because each packet delivered in simplePack carries more information elements due to the higher packing ratio.

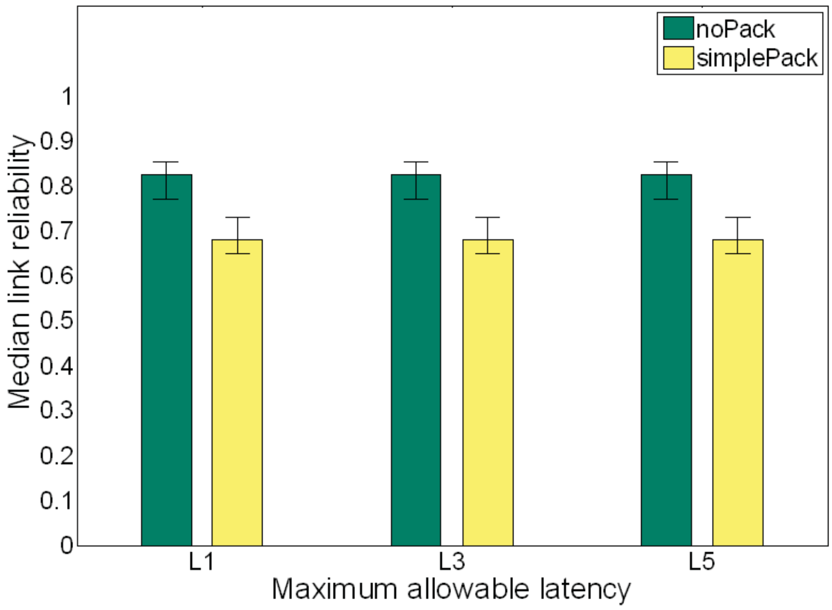

Figure 28: Link reliability: $D6$

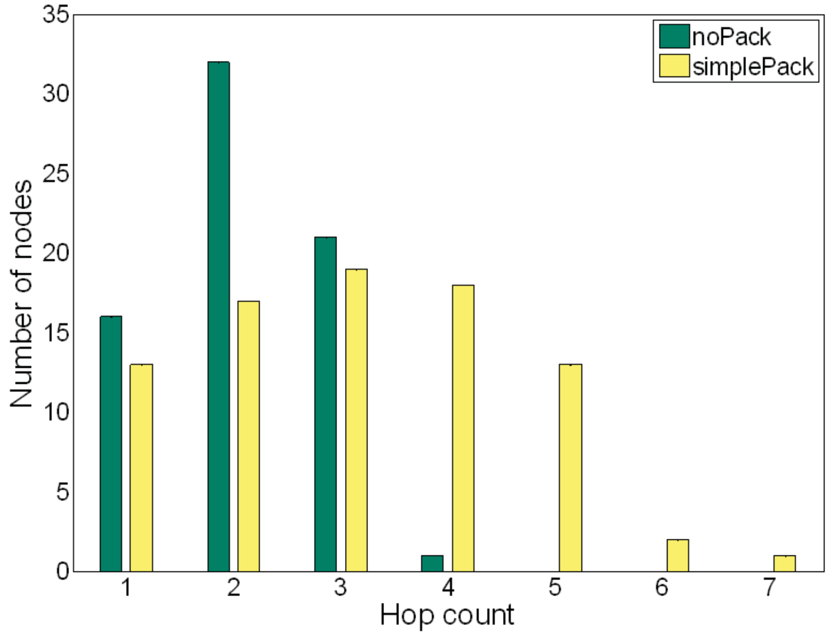

Figure 29: Histogram of routing hop count: $D6$ with maximum latency $L1$

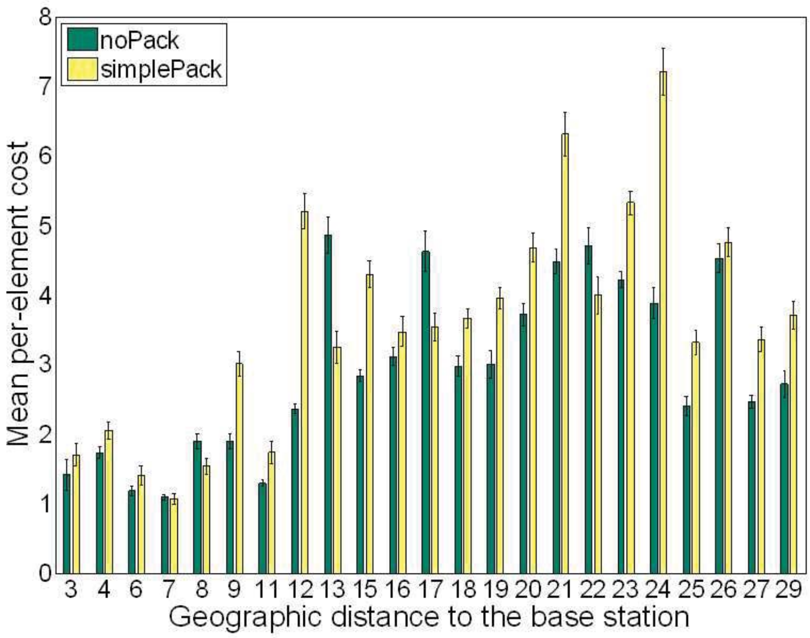

Figure 30: Per-element delivery cost vs. geo-distance: $D6$ with maximum latency $L1$

traffic patterns.) We see that, for most of the cases, the per-element delivery cost is higher in simplePack. Note that similar arguments explain why simplePack has higher delivery cost than noPack in traffic pattern $D9$ and why SL also has higher delivery cost than noPack in several cases (e.g., for traffic pattern $D6$-$L1$). In view with the consistently better performance in tPack, these observations demonstrate again the importance of considering network conditions and data aggregation constraints in in-network processing.

Event Traffic

Figures 31-35 show the measurement results for event traffic pattern E_{lites}. The overall trend on the relative protocol performance is similar to that in the periodic traffic patterns $D3$, $D6$, and $D9$. Even though the delivery reliability tends to be high for all protocols, tPack still achieves lower delivery cost and latency jitter, as well as 100% deadline catching ratio.

Related work

In-network processing (INP) has been well studied in sensornets, and many INP methods have been proposed for query processing [54, 69, 55, 58, 15, 14, 49, 30] and general data

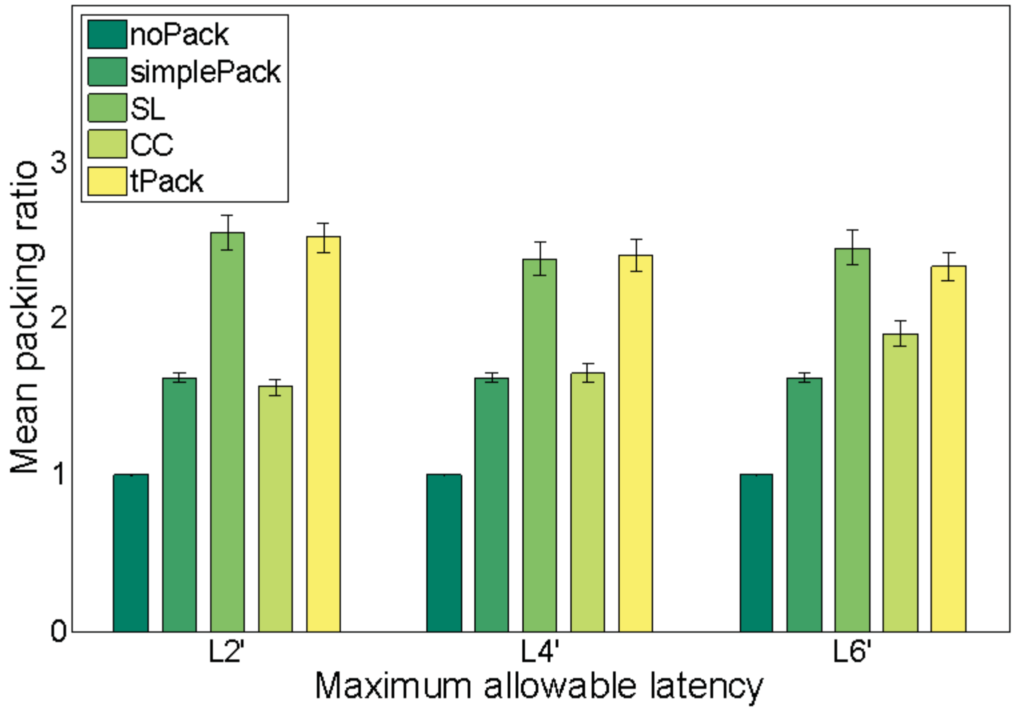

Figure 31: Packing ratio: E_{lites}

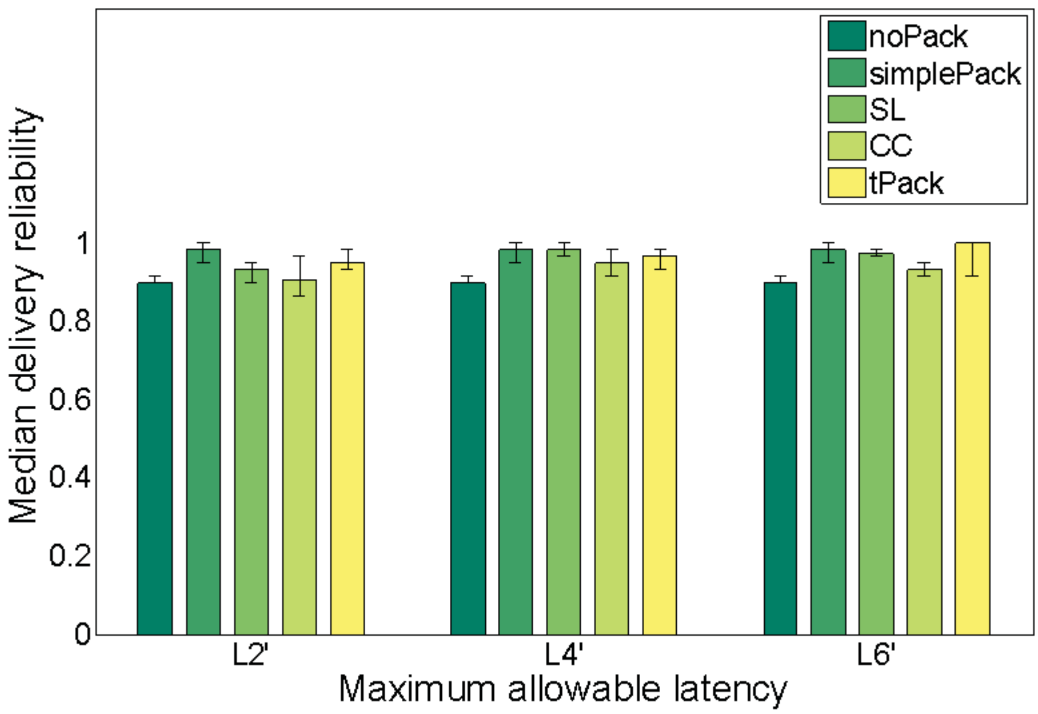

Figure 32: Delivery reliability: E_{lites}

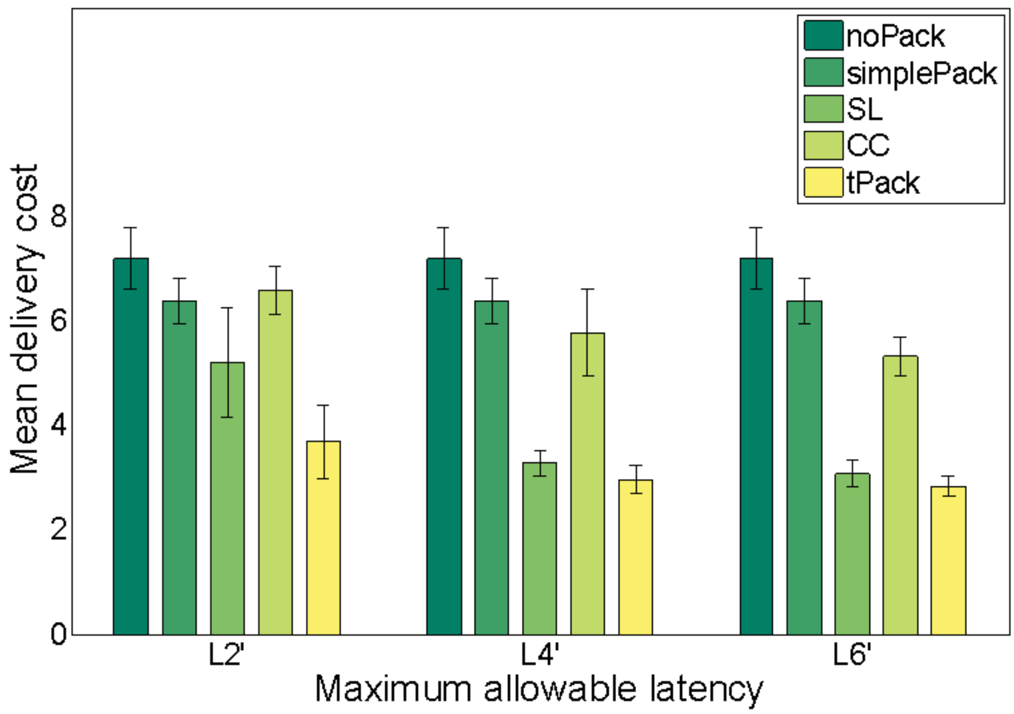

Figure 33: Delivery cost: E_{lites}

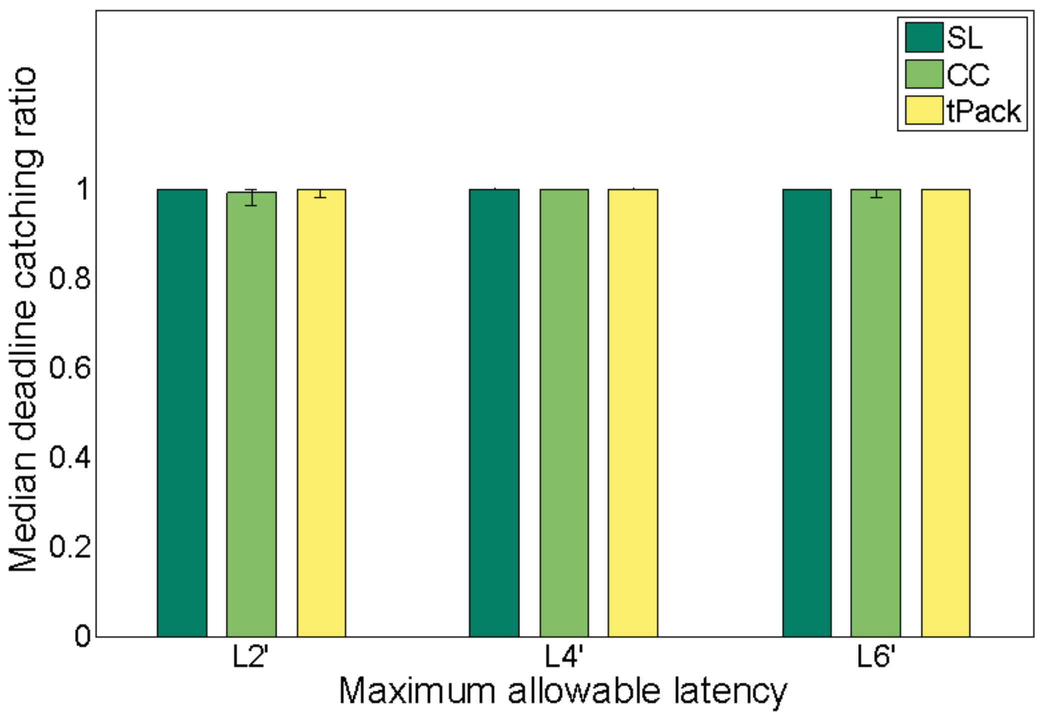

Figure 34: Deadline catching ratio: E_{lites}

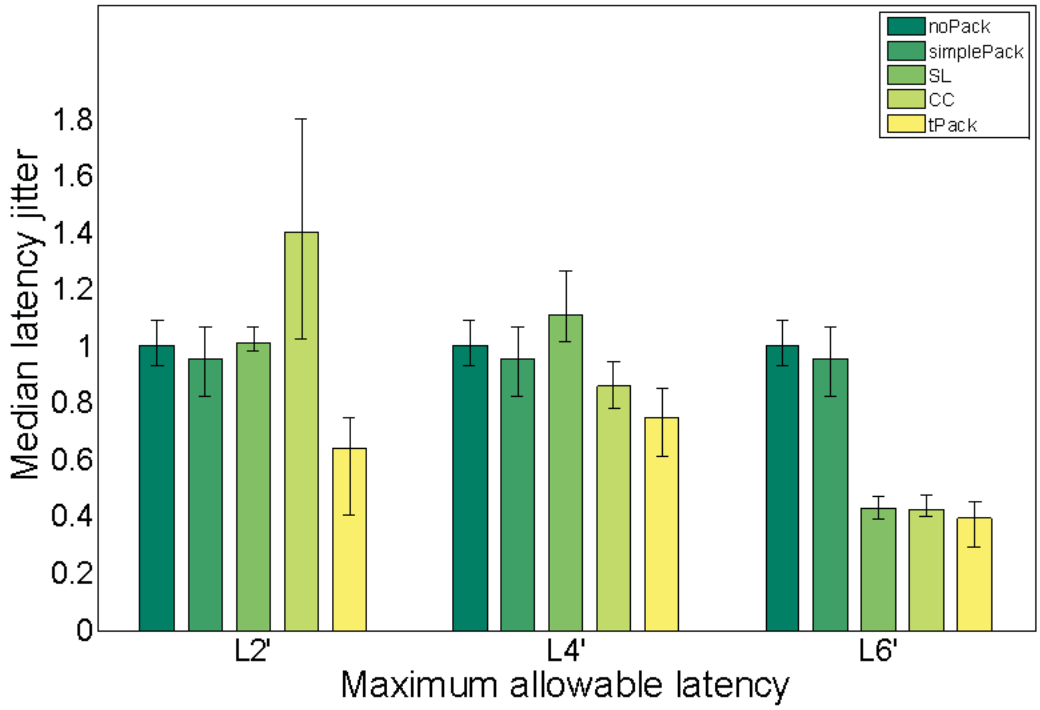

Figure 35: Latency jitter: E_{lites}

collection [20, 21, 43, 52, 61, 71]. When controlling spatial and temporal data flow to enhance INP, however, these methods did not consider application requirements on the timeliness of data delivery. As a first step toward understanding the interaction between INP and application QoS requirements, our study has shown the benefits as well as the challenges of jointly optimizing INP and QoS from the perspective of packet packing. As sensornets are increasingly being deployed for mission-critical tasks, it becomes important to address the impact of QoS requirements on general INP methods other than packet packing, which opens interesting avenues for further research.

As a special INP method, packet packing has also been studied for sensornets as well as general wireless and wired networks, where mechanisms have been proposed to adjust the degree of packet packing according to network congestion level [29, 35], to address MAC/link issues related to packet packing [48, 53, 46], to enable IP level packet packing [40], and to pack periodic data frames in automotive applications [62]. These works have focused on issues in local, one-hop networks without considering requirements on maximum end-to-end packet delivery latency in multi-hop networks. With the exception of [62], these works did not focus on scheduling packet transmissions to improve the degree of packet packing, and they have not studied the impact of finite packet size either. Saket et al. [62] studied packet packing in single-hop controller-area-networks (CAN) with finite packet size. Our work addresses the open questions on the complexity and protocol design issues for jointly optimizing packet packing and data delivery timeliness in multi-hop wireless sensornets.

Most closely related to our work are [10] where the authors studied the issue of optimizing INP under the constraint of end-to-end data delivery latency. But these studies did not consider aggregation constraints and instead assumed *total aggregation* where any arbitrary number of information elements can be aggregated into one single packet. These studies did not evaluate the impact of joint optimization on data delivery performance either. Our work focuses on settings where packet size is finite, and we show that aggregation constraints (in particular, maximum packet size and re-aggregation tolerance) significantly affect the problem complexity and protocol design. Using a high-fidelity sensornet testbed, we also systematically examine the impact of joint optimization on packet delivery performance in multi-hop wireless networks. By showing that tPack performs better than the algorithm SL and CC [10], our testbed based measurement results also demonstrate the benefits of considering realistic aggregation constraints in the joint optimization.

Solis et al. [63] also considered the impact that the timing of packet transmission has on data aggregation, and the problem of minimizing the sum of data transmission cost and delay cost has been considered in [59] and [38]. These studies also assumed total aggregation, and they did not consider hard real-time requirements on maximum end-to-end data delivery latency. Ye et al. [70] considered the local optimal stopping rule for data sampling and transmission in distributed data aggregation. It did not consider hard real-time requirement either, and it did not study network-wide coordination and the limit of data aggregation. Yu et al. [72] studied the latency-energy tradeoff in sensornet data gathering by adapting radio transmission rate; it did not study the issue of scheduling data transmission to improve the degree of data aggregation.

Concluding remarks

Through both theoretical and experimental analysis, we examine the complexity and impact of jointly optimizing packet packing and the timeliness of data delivery. We find that aggregation constraints (in particular, maximum packet size and re-aggregation tolerance) affect the problem complexity more than network and traffic properties do, which suggest the importance of considering aggregation constraints in the joint optimization. We identify conditions for the joint optimization to be strong NP-hard and conditions for it to be solvable in polynomial time. For cases when it is polynomial-time solvable, we solve the problem by transforming it to the maximum weighted matching problem in interval graphs; for cases when it is strong NP-hard, we prove that there is no polynomial-time approximation scheme (PTAS) for the problem. We also develop a local, distributed online protocol tPack for maximizing the local utility of each node, and we prove the competitiveness of the protocol with respect to optimal solutions. Our testbed-based measurement study also corroborates the importance of QoS- and aggregation-constraint aware optimization of packet packing.

While this chapter has extensively studied the complexity, algorithm design, and impact of jointly optimizing packet packing and data delivery timeliness, there are still a rich set of open problems. Even though we have analyzed the competitiveness of tPack for non-trivial scenarios and this has given us insight into the behavior of tPack, it remains an open question on how to characterize in a closed form the competitiveness of tPack and non-oblivious online algorithms in broader contexts. The analytical and algorithmic design mechanisms developed for packet packing may well be extensible to address other in-network processing methods such as data fusion, and a detailed study of this will help us better understand the structure of the joint optimization problem and will be interesting future work to pursue. We have focused on the scheduling aspect of the joint optimization, and we are able to use mathematical tools such as interval graphs to model the problem; on the other hand, how to mathematically model and analyze the impact of the joint optimization on spatial data flow is still an open question and is beyond the scope of most existing network flow theory, thus it will be interesting to explore new approaches to modeling and solving the joint optimization problem.

CHAPTER 3
ENERGY-EFFICIENT NETWORK CODING BASED ROUTING

Preliminary

Ahlswede *et al.* [6] first proposed the network coding technique. The authors showed that the use of network coding can effectively increase throughput in wired networks. Since then, different network coding strategies have been studied, e.g., linear network coding [6], non-linear network coding [45] and random network coding [31] [34]. Ho et al. [31] proved that the use of random network coding can achieve the theoretical maximal throughput in wireless networks. And Eryilmaz *et al.* [18] show that network coding can reduce transmission latency, therefore can increase the throughput in multicast traffic flow.

Recently, Chachulski *et al.* [13] propose MORE, the first protocol to integrate random network coding with opportunistic routing for unicast flow in wireless mesh networks. Experiment results show that MORE yields a higher network throughput than ExOR [12] which only uses opportunistic routing. Based on the framework of [13], [50] [41] further improve the network throughput by introducing different ACK and rate control schemes. To the best of our knowledge, however, there has been no systemic study on spatial and energy consumption control on network-coding-based (NC-based) routing, which is of great importance in power-constrained distributed systems, e.g., wireless sensor networks.

In this work, we study the open problem of minimal cost NC-based routing in wireless networks. Our main contributions are as follows.

- We propose an effective load based approach to measure the expected number of transmissions of NC-based transmission for arbitrary topologies. This is the first mathematical framework to compute the transmission cost of NC-based routing.

- We propose a polynomial greedy algorithm to compute the minimal transmission cost and the corresponding routing braid for NC-based routing. We prove the optimality of this algorithm and an upper bound of transmission cost for the optimal NC-based routing braid, which is equal to the cost of shortest single path routing.

- Based on the algorithm we proposed, we design and implement EENCR, an energy-efficient NC-based routing protocol, for resource-constrained sensor platforms. In EENCR, we incorporate the 4-bit link estimator of CTP [27], realize a light-weight distributed implementation for our greedy forwarder set selection algorithm in the rouging engine, and design a modified null-spaced-based (M-NSB) coded feedback scheme and a corresponding rate control component. Compared to CTP, EENCR introduces zero additional communication cost but yields an optimal routing braid with lower cost than the shortest single path routing.

- We evaluate the performance of EENCR on the NetEye testbed by comparing it with CTP [27], MORE [13] and CodeOR [50]. Experiment results show that EENCR achieves a close to 100% reliability with a large transmission cost reduction of CTP, i.e., 25 - 28%. And EENCR further improves the goodput of NC-based routing protocol by adaptively selecting the forwarders instead of utilizing the whole forwarder candidate set.

The remaining of this chapter is organized as follows. We first introduce the system settings and problem definition. We then propose the effective-load-based framework to compute the transmission cost of NC-based routing. Based on this framework, we design a polynomial-time greedy algorithm that can compute the optimal routing braid for arbitrary topologies. Next we present EENCR, which includes a distributed implementation of our greedy algorithm. We evaluate the performance of EENCR under different topologies on the NetEye testbed. Before we conclude this chapter, we discuss related work in the field of network coding.

System settings and problem definition

In this section, we first present the system settings we used in this study. Next we explain why we choose intra-flow network coding in designing efficient routing protocol for mission-critical WCPS. Based on the system model, we formally define the problems of transmission cost computation and optimization for NC-based routing.

System settings

In this study, we model a wireless network as a graph $G = (V, E)$ with node S as the source and T as the destination. For each node $i \in G$, we use U_i and D_i to denote the set of senders and receivers of i, respectively. And we denote the forwarder set of i as $FS_i \subset D_i$. For each link $i \to j \in E$, we denote ETX_{ij}^x as its expected number of transmission to deliver a packet with length x and $P_{ij}^x = \frac{1}{ETX_{ij}}$ as the corresponding link reliability. Since network coding will not change the packet length during the transmission, we use ETX_{ij} and P_{ij} for simplicity. Then we define $C_{iT}(x)$ as the transmission cost of delivering x linear independent packets from i to T, and $C_{iD_i}(x)$ as the expected number of broadcasts of node i when nodes in D_i collectively receive x linear independent coded packets from i. Assuming S needs to deliver K packets as a batch to T, we define K_i^j as the number of linear independent packets node i received from node j.

Comparison between inter-flow and intra-flow network coding

In studies of network coding in wireless environment, there are two general techniques. The first one is inter-flow network coding, which allows a node to encode packets from different flows and broadcast the coded packet. The second one is intra-flow network coding, which allows a node only encode packets of the same flow and broadcast the coded packet. Both techniques have their advantages and suitable application scenarios. Inter-flow network coding suits well for multiple source-destination pairs in a wireless environment while intra-flow network coding suits better for single unicast data flow in wireless networks. In mission-critical WCPS, the most common traffic flow is called convergecast, in which multiple sources send data to single destination. Convergecast is a variant of both multiple source-destination flows and single unicast flow. Theoretically, we can use both inter-flow and intra-flow technique to improve the energy efficiency of convergecast in WCPS. However, inter-flow coding requires perfect feedback information for a sending node to make encoding decision. This requirement incurs very high communication overhead and therefore makes inter-flow network coding unfavorable for convergecast applications in WCPS. Furthermore, applying inter-flow coding in convergecast also requires defining auxiliary nodes in the network, which will make the whole solution too complex for mission-critical WCPS. On the contrary, intra-flow coding requires little feedback information. Not only will this feature of intra-flow coding makes it suitable for efficiency improvement in WCPS, but also it will make the protocol design and implementation simpler for resource-constrained sensor platforms.

Problem definition

We define the minimal cost NC-based routing problem as follows:

Problem $\mathbf{Q_0}$ Given a graph $G = (V, E)$ with one source S and one destination T, find the optimal total transmission cost and the corresponding FS_i for each node i to deliver K packets using intra-flow random network coding from S to T.

To the best of our knowledge, however, there has been no study on how to measure the transmission cost of intra-flow network coding, letting alone the optimal transmission cost. Therefore, we need to first find a way to measure the transmission cost of NC-based routing before we can solve $\mathbf{Q_0}$. Therefore, we define the following problem:

Problem $\mathbf{Q_1}$ the same as $\mathbf{Q_0}$ except that $FS_i = D_i$ for each node i.

The solution to problem $\mathbf{Q_1}$ can provide a mathematical framework to compute the expected transmission cost of NC-based routing. Not only will this framework provide a tool for our solution to problem Q_0, but also it will shed lights towards energy-efficiency study of NC-based transmission in future study. In the following sections, we will propose optimal polynomial-time algorithms for both problem $\mathbf{Q_1}$ and $\mathbf{Q_0}$.

Cost optimization for NC-based routing

In this section, we first propose an effective load assignment algorithm to solve problem $\mathbf{Q_1}$. The key idea of this algorithm is to compute the number of encoded packets each intermediate node should forward and the corresponding cost. Based on this approach, we then design a distributed polynomial-time algorithm to optimally solve $\mathbf{Q_0}$. For each node i in $\mathbf{Q_0}$, we choose the forwarder set FS_i out of D_i using a greedy algorithm based on the transmission cost from each node in D_i to T We prove this algorithm's optimality and show that the optimal transmission cost of NC-based routing has an upper bound that equals to the transmission cost of the shortest path routing.

Effective load based assignment algorithm for $\mathbf{Q_1}$

In NC-based opportunistic routing protocols, such as MORE [13], the network throughput is significantly improved compared with single path routing. However, the transmission cost of these protocols are not carefully controlled and it may be higher than the cost of single path routing since every intermediate node will forwards re-encoded packets to its own forwarder candidate set. To precisely measure and control the transmission cost while still fully utilizing the benefit of network coding on throughput, we propose a concept called *effective load*.

Definition 1 *For a node j in the forwarder candidate set FCS_i, the* ***effective load*** *L_j is defined as the number of linear independent packets that are received by j but not by any of the other nodes in FCS_i that has lower transmission cost to the destination.*

To demonstrate this concept, we first look at the following example in Figure 36. In this example, the source node S has $K = 3$ packets that needs delivering to T and $C_{AT} < C_{BT}$. Therefore, node A has a higher priority than B in FCS_S. When S stops broadcasting, the coding vectors of packets received by node A are $\{1, 2, 3\}$ and $\{1, 1, 1\}$ and the vectors at node B are $\{2, 3, 5\}$ and $\{1, 1, 1\}$. Since node A has a lower transmission cost to T than B, node A has an effective load $L_A = 2$. Node B only has an effective load $L_B = 1$ because the packet with coding coefficient $\{1, 1, 1\}$ is also received by A. If both node A and B forward up to their effective load of re-encoded packets to T, T will receive 3 linear independent packets, which is just enough to decode the whole batch. In the meantime, there will be no unnecessary re-encoding forwarding operations from FSC_S to T.

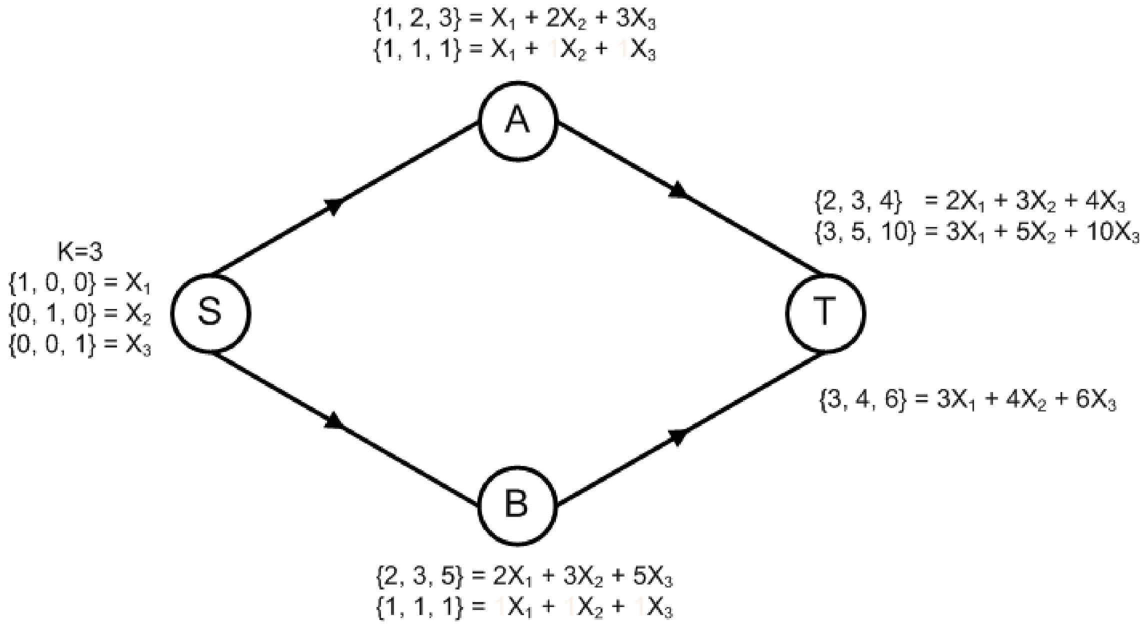

Figure 36: An illustrating example of NC-based routing

Based on the concept of effective load, we then propose a framework to compute the transmission cost of NC-based routing based on different effective load between nodes within the same forwarder candidate set, i.e., given a node i, each node $j \in D_i$ will forward L_j linear independent packets to the destination

To better illustrate how to use the effective load approach to compute the transmission cost of NC-based routing, we first study the following example in Figure 37.

In this diamond topology, we define that $P_2 \geq P_4 \geq P_6$. The whole transmission process can be divided into two steps. The first step is node S broadcasting to $D_S = \{A, B, C\}$ and the second step is nodes in D_S forwarding re-encoded packets to T. In the first step, we treat node A, B, C as one single virtual node V_{D_S}. The link reliability of link $S \rightarrow V_{D_S}$ is then expressed as $P_{SV_{D_S}} = 1 - (1 - P_1)(1 - P_3)(1 - P_5)$. Therefore, the transmission cost for the first step is

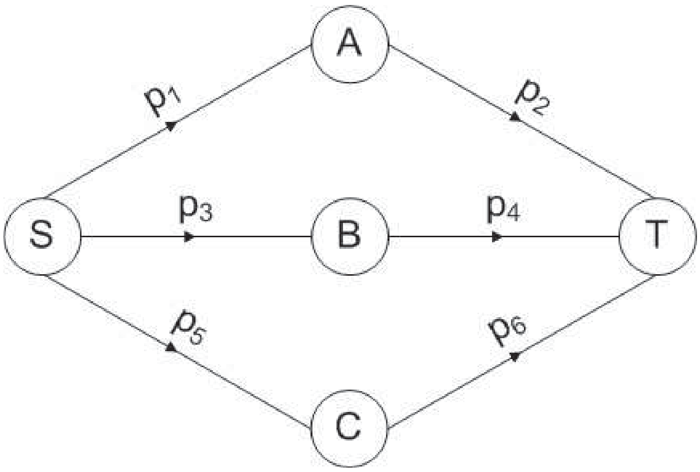

Figure 37: Example topology

$$C_{SD_S}(K) = \frac{K}{P_{SV_{D_S}}} = \frac{K}{1-(1-P_1)(1-P_3)(1-P_5)} \tag{34}$$

In the second step, since we have $P_2 \geq P_4 \geq P_6$, we want path $A \rightarrow T$ to forward as many packets as it is capable of and path $C \rightarrow T$ to forward as least packets as needed. To compute the effective load for nodes inD_S, we first compute K_i, the expected number of linear independent packets received by each node from S in the first step.

$$\begin{cases} K_A^S = & \dfrac{KP_1}{1-(1-P_1)(1-P_3)(1-P_5)} \\ K_B^S = & \dfrac{KP_3}{1-(1-P_1)(1-P_3)(1-P_5)} \\ K_C^S = & \dfrac{KP_5}{1-(1-P_1)(1-P_3)(1-P_5)} \end{cases} \tag{35}$$

Using L_i to denote the number of linear independent packets node i needs to forward to T, it is easy to see that $L_A = K_A^S$. However, we cannot simply calculate L_B as $min(K - L_A, K_B^S)$ because node B and A may receive some same packets, resulting in less entropy held by B. Instead, we need to compute $K_B^{S'}$, the expected number of linear independent packets that are received by node B but not A.

$$K_B^{S'} = K\frac{K_B^S}{K}(1-P_1) = K_B^S(1-P_1) \tag{36}$$

The detailed deduction to compute $K_B^{S'}$ is to solve an easy probability theory problem and is hence omitted. It is easy to see that $K_B^{S'} < K - L_A$, thus we have $L_B = K_B S'$. Similarly, we have $L_C = K_C^{S'} = K_C^S(1-P_1)(1-P_3)$ and we can verify that $L_A + L_B + L_C = K$. Combining these intermediate results, we have the total transmission cost computed as:

$$
\begin{aligned}
C_S(K) &= C_{SD_S}(K) + C_{AT}(L_A) \\
&\quad + C_{BT}(L_B) + C_{CT}(L_C) \\
&= \frac{K}{1-(1-P_1)(1-P_3)(1-P_5)} \\
&\quad + \frac{L_A}{P_2} + \frac{L_B}{P_4} + \frac{L_C}{P_6} \\
&= \frac{K}{1-(1-P_1)(1-P_3)(1-P_5)} \\
&\quad [1 + \frac{P_1}{P_2} + \frac{P_3(1-P_1)}{P_4} + \frac{P_5(1-P_1)(1-P_3)}{P_6}]
\end{aligned} \tag{37}
$$

Through this example, we demonstrate how to compute the transmission cost of NC-based routing. The basic idea is to first compute the broadcast cost treating all the nodes in the forwarder candidate set as one single virtual node, and then compute the effective load, i.e., the number of re-encoded packets needs to be forwarded at each node in the forwarder candidate set based on an non-decreasing order of their cost to the destination.

Although the topology in Figure 37 consists of only node-disjoint paths from the source to the destination, we can generalize this approach to recursively compute the cost of NC-based routing in arbitrary topologies. We formally present this computing process as Algorithm 2. Basically, each node i runs Algorithm 2 to compute its transmission cost to the destination if every node in D_i has its transmission cost computed and updated. Node i then sends its own cost information to its sender(s). The sender(s) then run this algorithm again to compute their transmission cost to the destination. By the end of this backwards recursive process, the source node S will be able to compute its transmission cost to the destination based on the transmission cost of nodes in D_S. Note that the complexity of Algorithm 2 is $O(|V| \lg |V|)$, which makes it suitable for power-constrained computation platforms, e.g., the Telosb sensor platform.

Algorithm 2 Compute the transmission cost of NC-based routing for the current node S with M forwarder candidates

Input: current node S, $D_S = \{A_1, A_2, \ldots, A_M\}$
Output: $C_S(1)$: the expected number of transmissions to deliver 1 packet from S to T
Sort nodes in D_S by a non-descending order of $C_{A_i}(1)$, where $i = 1, 2, \ldots, M$.
Sorted nodes are labeled as $\{A'_1, A'_2, \ldots, A'_M\}$
$C_{SD_S}(1) = \frac{1}{1-\prod_{i=1}^{M}(1-P_{SA'_i})}$
$L_{A'_1} = C_{SD_S}(1)P_{SA'_1}$
$F = 1 - P_{SA'_1}$
for $i \to 2, 3, \ldots, M$ **do**
    $L_{A'_i} = C_{SD_S}(1)P_{SA'_i}F$
    $C_{A'_i}(L_{A'_i}) = L_{A'_i}C_{A'_i}(1)$
    $F = F(1 - P_{SA'_i})$
end for
$C_S(1) = C_{SD_S}(1) + \sum_{i=1}^{M} C_{A'_i}(L_{A'_i})$

As described above, the major principle we use here is to always assign more traffic load to forwarders with lower cost, which implies that we should apply different utilization of forwarders in the DAG to minimize the transmission cost instead of fully utilizing every possible path in the network. This observation provides two insights: 1)it shows that we do not need full network coding redundancy in the network to perform regular data transmission, which would cause higher transmission cost and contention; 2) extra redundancy may be used to provide proactive protection to mission-critical networks against single node failures. These two insights lead us to the solution to problem $\mathbf{Q_0}$ in this chapter and problem $\mathbf{Q}$ in Chapter 4.

Optimal NC-based transmission cost algorithm

In the last section, we proposed a distributed algorithm executed by each node to compute the transmission cost of NC-based routing from a given source to the destination. However, there still lacks a precise control on transmission cost in NC-based routing, making NC-based transmission energy-inefficient. This energy-inefficiency is especially severe in dense networks where each node has many forwarder candidates.

In MORE-based protocols, forwarder candidates with low expected effective load are usually not allowed to forward the flow to reduce the contention in the network, which can reduce the transmission cost sometimes. However, this reduction is not guaranteed and sometimes it may even increase the transmission cost. Based on the observations we had a priori, we design a distributed greedy algorithm, Algorithm 3. The basic idea of Algorithm 3 is as follows. For an input node S, we first sort all nodes in D_S in a non-descending order of their transmission cost to the destination. We then remove the node A'_i with the lowest transmission cost from the sorted D_S, add it to the forwarder set FS_S and compute the total transmission cost using Algorithm 2. If the transmission cost of S can be reduced by adding A'_i to FS_S, we keep it in FS_S and add another node with the lowest cost from the remaining sorted D_S. We continue this loop until either of the following two conditions is satisfied:

1. the sorted D_S is empty, i.e., all receivers of S have been selected into the FS_S;
2. moving another node from the sorted D_S to FS_S would increase the total transmission cost from S to T.

Each non-destination node executes this algorithm to determine the minimal transmission cost from itself to the destination T and the corresponding forwarder candidate set. Upon the convergence of the whole network, we will get the solution to problem $\mathbf{Q_0}$.

Algorithm 3 Compute the minimal transmission cost of NC-based routing and the corresponding FCS for the input node S with M forwarders

Input: node S, $D_S = \{A_1, A_2, \ldots, A_M\}$, $FS_S = \emptyset$
Output: $C^*_S(1)$: the minimal transmission cost to deliver 1 packet from S to T
Sort nodes in D_S by a non-descending order of $C_{A_i}(1)$, where $i = 1, 2, \ldots, M$.
Sorted nodes are labeled as $\{A'_1, A'_2, \ldots, A'_M\}$
$FS_S = \{A'_1\}$
$C^*_S(1) = \frac{1}{P_{SA'_1}} + C_{A'_1}(1)$
for $i \to 2, 3, \ldots, M$ **do**
  Run Algorithm 2 with input S and $D_S = \{A'_1, \ldots, A'_i\}$
  Get the result as $C^{new}_S(1)$
  if $C^{new}_S(1) > C_S(1)$ **then**
    break
  else
    $FS_S = FS_S \cup A'_i$
    $C^*_S(1) = C^{new}_S(1)$
  end if
end for

The complexity of this algorithm is $O(|V|^2 \lg |V|)$. In NC-based routing, the size of FC_S can also be one and in this case the routing braid is the equivalent to the shortest single path. Next we show readers the optimality of this algorithm by proving the following theorem:

Theorem 9 *Given a node S and its forwarder candidate set $D_S = \{A_1, A_2, \ldots, A_M\}$, Algorithm 3 yields the minimal transmission cost to the destination node of NC-based routing and the corresponding forwarder set.*

Proof We prove the correctness of this theorem by contradiction. Given a node S and its forwarder candidate set D_S, we denote the minimal transmission cost as C^* and the corresponding transmission forwarder set is FS_S^* with a cardinality of k. We sort nodes in FS_S^* in non-descending order of their transmission cost to the destination and denote them as $FS_S^* = \{A_1^*, A_2^*, \ldots, A_k^*\}$ where $C_{A_1^*} \le C_{A_2^*} \le, \ldots, \le C_{A_k^*}$.

If this theorem is not correct, then there exists at least one node A_x having $C_{A_x} \le C_{A_i^*}$ for some integer $i \in [1, k]$. Without loss of generality, we assume that $C_{A_{k-1}^*} \le C_{A_x} \le C_{A_k^*}$. We will have a contradiction when we can find a forwarder set FS_S^{**} that has a lower transmission cost C^{**} than C^*. To find this contradiction, we study the following forwarder sets:

$$\begin{aligned} FS_S^* &= \{A_1^*, A_2^*, \ldots, A_k^*\} \\ FS_S^1 &= FS_S^* - \{A_k^*\} \\ FS_S^2 &= FS_S^* \cup \{A_x\} \end{aligned} \tag{38}$$

For each forwarder set, we compute the transmission cost for these forwarder sets using Algorithm 2. The transmission cost of FS_S^* is expressed as:

$$C^* = \frac{1 + \sum_{i=1}^{k} [C_{A_i^*} P_{SA_i^*} \prod_{j=1}^{i-1} (1 - P_{SA_j^*})]}{1 - \prod_{i=1}^{k} (1 - P_{SA_i^*})} \tag{39}$$

Compared with FS_S^*, FS_S^1 does not have node A_k^*, therefore cost C_1 is expressed as:

$$C_1 = \frac{1 + \sum_{i=1}^{k-1} [C_{A_i^*} P_{SA_i^*} \prod_{j=1}^{i-1} (1 - P_{SA_j^*})]}{1 - \prod_{i=1}^{k-1} (1 - P_{SA_i^*})} \tag{40}$$

On the other hand, FS_S^2 consists of both FS_S^* and node A_x. Since A_x has a lower transmission cost than node A_k^*, we compute C_2 as:

$$
\begin{aligned}
C_2 \quad = \quad & \frac{1}{1-(1-P_{SA_x})\prod_{i=1}^{k}(1-P_{SA_i^*})} \\
& \cdot\{1+\sum_{i=1}^{k-1}[C_{A_i^*}P_{SA_i^*}\prod_{j=1}^{i-1}(1-P_{SA_j^*})]+C_{A_x}P_{SA_x}\prod_{i=1}^{k-1}(1-P_{SA_i^*}) \\
& +P_{SA_k^*}(1-P_{SA_x})\prod_{i=1}^{k-1}(1-P_{SA_i^*})\}
\end{aligned}
\tag{41}
$$

Based on our assumption, C^*, C_1 and C_2 have the following relations:

$$
\begin{aligned}
C^*-C_1 &\leq 0 \\
C^*-C_2 &\leq 0
\end{aligned}
\tag{42}
$$

The basic idea next is to prove that $C^*-C_2 \geq 0$ when $C^*-C_1 \leq 0$, which leads to a contradiction.

$$
\begin{aligned}
& \tfrac{C^*}{C_1} = \frac{1-\prod_{i=1}^{k-1}(1-P_{SA_i^*})}{1-\prod_{i=1}^{k}(1-P_{SA_i^*})}\cdot\frac{1+\sum_{i=1}^{k}[C_{A_i^*}P_{SA_i^*}\prod_{j=1}^{i-1}(1-P_{SA_j^*})]}{1+\sum_{i=1}^{k-1}[C_{A_i^*}P_{SA_i^*}\prod_{j=1}^{i-1}(1-P_{SA_j^*})]} \leq 1 \\
\Leftrightarrow \quad & [1-\prod_{i=1}^{k-1}(1-P_{SA_i^*})]\cdot\{1+\sum_{i=1}^{k}[C_{A_i^*}P_{SA_i^*}\prod_{j=1}^{i-1}(1-P_{SA_j^*})]\} \\
& -[1-\prod_{i=1}^{k}(1-P_{SA_i^*})]\cdot\{1+\sum_{i=1}^{k-1}[C_{A_i^*}P_{SA_i^*}\prod_{j=1}^{i-1}(1-P_{SA_j^*})]\} \leq 0 \\
\Leftrightarrow \quad & \{1+\sum_{i=1}^{k-1}[C_{A_i^*}P_{SA_i^*}\prod_{j=1}^{i-1}(1-P_{SA_j^*})]\}\cdot\{\prod_{i=1}^{k}(1-P_{SA_i^*})-\prod_{i=1}^{k-1}(1-P_{SA_i^*})\} \\
& +[1-\prod_{i=1}^{k-1}(1-P_{SA_i^*})]P_{SA_k^*}C_{A_k^*}\prod_{i=1}^{k-1}(1-P_{SA_i^*}) \leq 0 \\
\Leftrightarrow \quad & \{1+\sum_{i=1}^{k-1}[C_{A_i^*}P_{SA_i^*}\prod_{j=1}^{i-1}(1-P_{SA_j^*})]\}\cdot(-P_{SA_k^*}) \\
& +[1-\prod_{i=1}^{k-1}(1-P_{SA_i^*})]P_{SA_k^*}C_{A_k^*} \leq 0 \\
\Leftrightarrow \quad & \\
& 1+\sum_{i=1}^{k-1}[C_{A_i^*}P_{SA_i^*}\prod_{j=1}^{i-1}(1-P_{SA_j^*})] \geq [1-\prod_{i=1}^{k-1}(1-P_{SA_i^*})]C_{A_k^*}
\end{aligned}
\tag{43}
$$

To accomplish this goal, we conduct some mathematical transformation of two inequities above. The first inequity is between C^* and C_1. Starting from the fact that $\frac{C^*}{C_1} \leq 1$, we have a useful result in Equation 43:

And the second result is between C^* and C_2 and this time we directly expand the difference:

$$
\begin{aligned}
C^* - C_2 &= \frac{1 + \sum_{i=1}^{k}[C_{A_i^*} P_{SA_i^*} \prod_{j=1}^{i-1}(1 - P_{SA_j^*})]}{1 - \prod_{i=1}^{k}(1 - P_{SA_i^*})} \\
&\quad - \frac{1}{1 - (1 - P_{SA_x}) \prod_{i=1}^{k}(1 - P_{SA_i^*})} \\
&\quad \cdot \{1 + \sum_{i=1}^{k-1}[C_{A_i^*} P_{SA_i^*} \prod_{j=1}^{i-1}(1 - P_{SA_j^*})] + C_{A_x} P_{SA_x} \prod_{i=1}^{k-1}(1 - P_{SA_i^*}) \\
&\quad + C_{A_k^*} P_{SA_k^*}(1 - P_{SA_x}) \prod_{i=1}^{k-1}(1 - P_{SA_i^*})\} \\
&= \{1 + \sum_{i=1}^{k-1}[C_{A_i^*} P_{SA_i^*} \prod_{j=1}^{i-1}(1 - P_{SA_j^*})]\} \\
&\quad \cdot \{\frac{1}{1 - \prod_{i=1}^{k}(1 - P_{SA_i^*})} - \frac{1}{1 - (1 - P_{SA_x}) \prod_{i=1}^{k}(1 - P_{SA_i^*})}\} \\
&\quad + \prod_{i=1}^{k-1}(1 - P_{SA_i^*}) \\
&\quad \cdot \{P_{SA_k^*} C_{A_k^*}[\frac{1}{1 - \prod_{i=1}^{k}(1 - P_{SA_i^*})} - \frac{1 - P_{SA_x}}{1 - (1 - P_{SA_x}) \prod_{i=1}^{k}(1 - P_{SA_i^*})}] \\
&\quad - P_{SA_x} C_{A_x} \cdot \frac{1}{1 - (1 - P_{SA_x}) \prod_{i=1}^{k}(1 - P_{SA_i^*})}\}
\end{aligned} \tag{44}
$$

Using some simple technique, we further transform the right-hand of Equation 44 and have the following result in Equation 45:

Using the result of Inequality 43 and the fact that $C_{A_k^*} > C_{A_x}$, we can find that the right hand side of Equation 45 is greater than 0, which means $C^* > C_2$ and shows the existence of a contradiction. And we note that using the above mathematical deduction framework, a contradiction can be found for any number of is where $i \in [1, k]$ and $C_{A_i^*} > C_{A_x}$. Therefore, we proved that we can find the minimal transmission cost of S to the destination by adding forwarder candidates with lower transmission cost to the destination into the forwarder set until adding more candidates will increase the C_S. By now, we complete our proof on the optimality of Algorithm 3 in computing the optimal NC-based routing topology.

$$
\begin{aligned}
C^* - C_2 &= \{1 + \textstyle\sum_{i=1}^{k-1}[C_{A_i^*} P_{SA_i^*} \prod_{j=1}^{i-1}(1 - P_{SA_j^*})]\} \\
&\quad \cdot \frac{P_{SA_x} \prod_{i=1}^{k}(1 - P_{SA_i^*})}{[1 - \prod_{i=1}^{k}(1 - P_{SA_i^*})][1 - (1 - P_{SA_x}) \prod_{i=1}^{k}(1 - P_{SA_i^*})]} \\
&\quad + \frac{P_{SA_x} \prod_{i=1}^{k-1}(1 - P_{SA_i^*})}{1 - (1 - P_{SA_x}) \prod_{i=1}^{k}(1 - P_{SA_i^*})} \cdot [\frac{P_{SA_k^*} C_{A_k^*}}{1 - \prod_{i=1}^{k}(1 - P_{SA_i^*})} - C_{A_x}] \\
&= \frac{P_{SA_x} \prod_{i=1}^{k-1}(1 - P_{SA_i^*})}{[1 - \prod_{i=1}^{k}(1 - P_{SA_i^*})][1 - (1 - P_{SA_x}) \prod_{i=1}^{k}(1 - P_{SA_i^*})]} \\
&\quad \cdot \{\{1 + \textstyle\sum_{i=1}^{k-1}[C_{A_i^*} P_{SA_i^*} \prod_{j=1}^{i-1}(1 - P_{SA_j^*})]\}(1 - P_{SA_k^*}) \\
&\quad + P_{SA_k^*} C_{A_k^*} - [1 - \textstyle\prod_{i=1}^{k}(1 - P_{SA_i^*})] C_{A_x}\} \\
&= \frac{P_{SA_x} \prod_{i=1}^{k-1}(1 - P_{SA_i^*})}{[1 - \prod_{i=1}^{k}(1 - P_{SA_i^*})][1 - (1 - P_{SA_x}) \prod_{i=1}^{k}(1 - P_{SA_i^*})]} \\
&\quad \cdot \{\{1 + \textstyle\sum_{i=1}^{k-1}[C_{A_i^*} P_{SA_i^*} \prod_{j=1}^{i-1}(1 - P_{SA_j^*})]\}(1 - P_{SA_k^*}) \\
&\quad + P_{SA_k^*} C_{A_k^*} - P_{SA_k^*} C_{A_x} - [1 - P_{SA_k^*} - \textstyle\prod_{i=1}^{k}(1 - P_{SA_i^*})] C_{A_x} \\
&= \frac{P_{SA_x} \prod_{i=1}^{k-1}(1 - P_{SA_i^*})}{[1 - \prod_{i=1}^{k}(1 - P_{SA_i^*})][1 - (1 - P_{SA_x}) \prod_{i=1}^{k}(1 - P_{SA_i^*})]} \\
&\quad \{(1 - P_{SA_k^*})\{1 + \textstyle\sum_{i=1}^{k-1}[C_{A_i^*} P_{SA_i^*} \prod_{j=1}^{i-1}(1 - P_{SA_j^*})] \\
&\quad - [1 - \textstyle\prod_{i=1}^{k}(1 - P_{SA_i^*})] C_{A_x}\} \\
&\quad + \cdot P_{SA_k^*} C_{A_k^*} - P_{SA_k^*} C_{A_x}\}
\end{aligned}
\tag{45}
$$

A theoretical comparison with other routing protocols

In the previous section, we proposed an optimal greedy algorithm that computes the minimal transmission cost of NC-based routing. Different from the heuristic control of spatial diversity in other MORE-based network coding opportunistic routing protocols, this algorithm intelligently explores the routing diversity in wireless transmission and only adds routes that can reduce the transmission cost into the forwarding topology. Therefore, our algorithm has a lower transmission cost than existing NC-based protocols [13] [41] [50]. When implementing a routing protocol, nonetheless, we still need to face the choice between NC-based routing and single path routing. In this section, we study a few properties of our solution, which demonstrates the advantage of our NC-based transmission algorithm over traditional single path routing in terms of energy efficiency, i.e., transmission cost.

In traditional single path routing, it is the common sense that we always want to select the shortest path in the network. The term "shortest" depends on different metrics or constraints we use, e.g., transmission cost, hop count, capacity and latency. However, when we use intra-flow network coding to tackle the forwarder selection problem in opportunistic routing to minimize the transmission cost, the first property we find for our solution is that the shortest (i.e., lowest cost) single path is not necessarily chosen into the transmission topology. This property is formally presented in the following theorem:

Theorem 10 *Given a node S with a candidate set FCS_S of M forwarders, the optimal forwarder set FS_S computed in Algorithm 3 does not always contain node A^* where $A^* \in FCS_S$ and $\frac{1}{P_{SA^*}} + C_{A^*} \leq \frac{1}{P_{SA_i}} + C_{A_i}$ for any $i \in FCS_S/\{A^*\}$.*

Proof The proof of this theorem is not complex. As long as we give an instance of node S with M forwarders that has the minimal cost transmission topology not including the lowest cost single path, we have the proof we need. Thus we build an instance in Figure 38.

In this instance, the lowest cost single path is $S \rightarrow A_3 \rightarrow T$ with a cost $\frac{1}{0.9} + \frac{1}{0.1} = 11.11$. After we run Algorithm 3, however, the optimal forwarder set we have is $FS_S = \{A_1, A_2\}$ because we have the following results:

$$\begin{aligned}
C_{\{A_1,A_2\}} &= \tfrac{1}{1-(1-0.1)(1-0.15)} \cdot [1 + \tfrac{0.1}{0.4} + \tfrac{0.15(1-0.1)}{0.2}] \\
&= \tfrac{1}{0.235} \cdot (1 + \tfrac{1}{4} + \tfrac{0.135}{0.2}) \\
&= 8.1915 \\
C_{\{A_1,A_2,A_3\}} &= \tfrac{1}{1-(1-0.1)(1-0.15)(1-0.9)} \cdot [1 + \tfrac{0.1}{0.4} + \tfrac{0.15(1-0.1)}{0.2} + \tfrac{0.9(1-0.1)(1-0.15)}{0.1}] \\
&= \tfrac{1}{0.9235} \cdot (1 + \tfrac{1}{4} + \tfrac{0.135}{0.2} + \tfrac{0.6885}{0.1}) \\
&= 9.5398 > C_{\{A_1,A_2\}}
\end{aligned} \tag{46}$$

Using this instance, we finish our proof for this theorem.

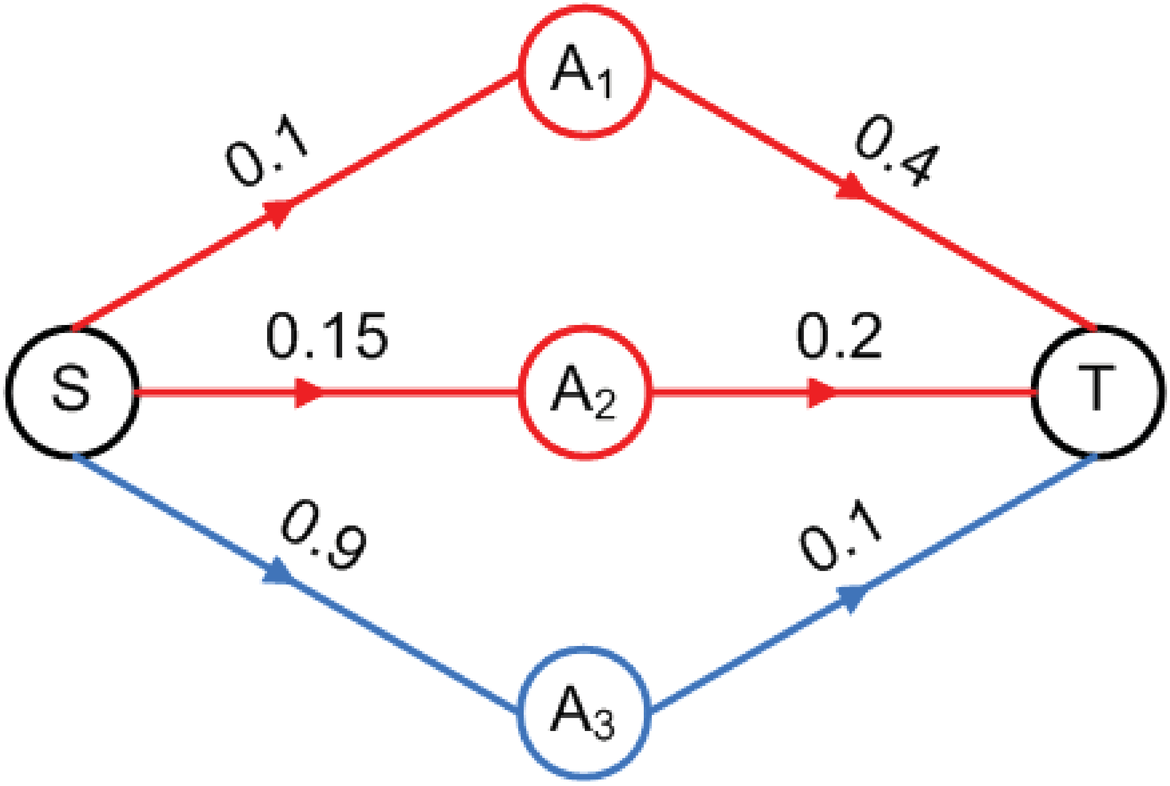

Figure 38: Routing braid v.s. single path routing

From this example, it is also easy to see that the optimal transmission cost of NC-based transmission is lower than that of shortest single path routing. This further raises the question: will the minimal cost of NC-based transmission always be better than that of single path routing? To answer this question, we propose the following theorem:

Theorem 11 *Given a node S with a candidate set FCS_S of M forwarders, the optimal transmission cost C_S^* computed in Algorithm 3 is always lower than or equal to $\frac{1}{P_{SA^*}+C_{A^*}}$ where $A^* \in FCS_S$ and $\frac{1}{P_{SA^*}} + C_{A^*} \leq \frac{1}{P_{SA_i}} + C_{A_i}$ for any $i \in FCS_S/\{A^*\}$.*

Proof Through Theorem 10 we showed that the forwarder on the lowest cost single path is not always in the forwarder set computed in Algorithm 3. Therefore, we prove the correctness of this theorem under two different cases:

1) $A^* \notin FS_S$ When the forwarder A^* on the lowest cost single path is not selected into FS_S, based on the greedy construction order of FS_S, we have the following inequity:

$$C_{A^*} \geq C_{A_j} \text{ for any } A_j \in FS_S \tag{47}$$

The only reason the algorithm does not add A^* into FS_S is because this operation will increase the total NC-based transmission cost. We denote $FS_S = \{A_1, A_2, \ldots, A_k\}$. This argument can be mathematically expressed as:

$$\begin{aligned}
C_S^* - C_{FS_S \cup \{A^*\}} &= \tfrac{1+\sum_{i=1}^{k}[C_{A_i}P_{SA_i}\prod_{j=1}^{i-1}(1-P_{SA_j})]}{1-\prod_{i=1}^{k}(1-P_{SA_i})} \\
&\quad -\tfrac{1+\sum_{i=1}^{k}[C_{A_i}P_{SA_i}\prod_{j=1}^{i-1}(1-P_{SA_j})]+C_{A^*}P_{SA^*}\prod_{i=1}^{k}(1-P_{SA_i^*})}{1-(1-P_{SA^*})\prod_{i=1}^{k}(1-P_{SA_i})} \\
&= \{1+\textstyle\sum_{i=1}^{k}[C_{A_i}P_{SA_i}\prod_{j=1}^{i-1}(1-P_{SA_j})]\} \\
&\quad \cdot\{\tfrac{1}{1-\prod_{i=1}^{k}(1-P_{SA_i})} - \tfrac{1}{1-(1-P_{SA^*})\prod_{i=1}^{k}(1-P_{SA_i})}\} \\
&\quad -\tfrac{C_{A^*}P_{SA^*}\prod_{i=1}^{k}(1-P_{SA_i^*})}{1-(1-P_{SA^*})\prod_{i=1}^{k}(1-P_{SA_i})} \\
&= \tfrac{\{1+\sum_{i=1}^{k}[C_{A_i}P_{SA_i}\prod_{j=1}^{i-1}(1-P_{SA_j})]\}P_{SA^*}\prod_{i=1}^{k}(1-P_{SA_i^*})}{\{1-\prod_{i=1}^{k}(1-P_{SA_i})\}\{1-(1-P_{SA^*})\prod_{i=1}^{k}(1-P_{SA_i})\}} \\
&\quad -C_{A^*}\cdot\tfrac{P_{SA^*}\prod_{i=1}^{k}(1-P_{SA_i^*})}{1-(1-P_{SA^*})\prod_{i=1}^{k}(1-P_{SA_i})} \\
&= \tfrac{P_{SA^*}\prod_{i=1}^{k}(1-P_{SA_i^*})}{1-(1-P_{SA^*})\prod_{i=1}^{k}(1-P_{SA_i})}\cdot\{\tfrac{1+\sum_{i=1}^{k}[C_{A_i}P_{SA_i}\prod_{j=1}^{i-1}(1-P_{SA_j})]}{1-\prod_{i=1}^{k}(1-P_{SA_i})} - C_{A^*}\} \\
&< 0
\end{aligned} \tag{48}$$

From this inequity, we then conduct the following transformation:

$$\begin{aligned}
&\tfrac{1+\sum_{i=1}^{k}[C_{A_i}P_{SA_i}\prod_{j=1}^{i-1}(1-P_{SA_j})]}{1-\prod_{i=1}^{k}(1-P_{SA_i})} - C_{A^*}\} < 0 \\
\Leftrightarrow\ & C_S^* - C_{A^*} < 0 \\
\Rightarrow\ & C_S^* < \tfrac{1}{P_{SA^*}} + C_{A^*}
\end{aligned} \tag{49}$$

Therefore, when A^* is not selected into FS_S, the optimal NC-based transmission cost C_S^* is lower than the transmission cost of shortest single path routing.

2) $A^* \in FS_S$ In this case, we consider three scenarios:

a) If $FS_S = \{A^*\}$, it is clear that $C_S^* = \frac{1}{P_{SA^*}} + C_{A^*}$.

b) If $FS_S \neq \{A^*\}$ and A^* is the first node selected into FS_S, $C^* < \frac{1}{P_{SA^*}} + C_{A^*}$ is implied in the greedy forwarder selection process of Algorithm 3.

c) If $FS_S \neq \{A^*\}$ and A^* is not the first node selected into FS_S, it is straightforward that

$$\frac{1}{1-(1-P_{SA^*})\prod_i^{i\in FS_S}(1-P_{SA_i})} > \frac{1}{P_{SA^*}} \tag{50}$$

And it is implied in the greedy forwarder selection process that before adding A^* into FS_S, C_{A^*} is greater than or equal to the forwarding cost from the old FS_S to the destination. Therefore we still have $C^* < \frac{1}{P_{SA^*}} + C_{A^*}$ under this scenario.

Combining all different scenarios, we can reach the conclusion that the minimal cost of NC-based transmission is always smaller than or equal to the shortest single path routing. This completes our proof.

Protocol design and implementation

After we proposed a minimal cost NC-based routing algorithm and proved its advantage over traditional shortest single path routing through theoretical analysis, we move on to deploy this algorithm into resource-constrained wireless platforms, e.g. wireless sensor networks. Not only do we need to implement this core algorithm, we also need other components to build the whole routing protocol. When designing a NC-based routing protocol, there are three key challenges, which are:

1. For each node, which neighbor of it should be selected into the forwarder set?
2. For each node, how many times of broadcast it should conduct for a batch before it stops?
3. For each node, how fast it should broadcast a re-encoded packet for a batch?

To address these challenges, we propose the energy-efficient NC-based routing (EENCR) protocol to perform minimal cost NC-based transmission in wireless sensor networks. EENCR is a fully distributed routing protocol that runs on every node in the network. In this section, we present three key components of EENCR, each of which addresses one of the challenges listed above.

Routing engine

Run on each node, the routing engine component computes the optimal forwarder set for the current node, which address the first challenge. We design the routing engine in EENCR based on the 4-bit link estimator component and routing engine component of the collection tree protocol(CTP). Our routing engine is responsible for the following assignments:

(a) Estimate the single link reliability from the current node to each of its 1-hop neighbor;

(b) Compute and update the minimal cost of NC-based transmission from the current node to the designated destination based on the received transmission cost information from its neighbors;

(c) Broadcast the computed minimal cost and the forwarder set effective load table to all its 1-hop neighbor;

(d) Provide the optimal effective load information to the ACK component and the rate control component.

The key difference between the routing engine in EENCR and CTP is that instead of selecting only the neighbor on the shortest single path as the next hop forwarder, EENCR selects a set of neighbors into the forwarder set using Algorithm 3 such that the total transmission cost can be further reduced. In this way, we make use of the routing diversity of wireless communication to the max extent.

Modified NSB coded feedback

The routing engine component decides the forwarder set for the current node. In NC-based routing, each node needs to know when it can stop broadcasting to its forwarders. The condition for a node i to stop broadcast is that nodes in the forwarder set of i have collectively receive L_i linear independent packets, where L_i is the effective load information computed from the routing engine component.

The usual way node i gets information to decide when to stop transmitting is via the ACK feedback from nodes in FS_i. One naive approach is to make nodes in FS_i transmit ACK on a per-packet basis. However, this per-packet ACK cannot be used in EENCR due to two reasons.

- The total size of per-packet ACK for the whole effective load is too large. In practical network coding protocols with symbol size $GF(2^8)$ and batch size 8, each coding vector contains 8 bytes. If a forwarder j wants to convey the whole coding vector space it received from i, it will need K_i^j 8-byte vectors, which is too large for energy-constrained sensor networks.

- Sending back per-packet ACKs will introduce high-contention and communication overhead in the network, which reduces the energy-efficiency of the whole protocol.

One approach to avoid this high overhead is to use coded feedback. First proposed in [60], the null-space-based (NSB) coded feedback scheme is originally designed to enhance reliability of an NC-based multicast protocol for multimedia applications in mobile ad hoc networks. To apply coded feedback into NC-based opportunistic routing, a Coded Cumulative ACK (CCACK) was proposed in [41]. CCACK designs a more complex ACK generating and testing scheme to solve the collective-space problem and false-positive problem when directly applying NSB in NC-based opportunistic routing. However, CCACK is designed to deploy in wireless mesh networks, where each node has a stronger computation power and larger storage space. It is hard to transplant it into sensor networks because:

- Compared to NSB, CCACK needs a much larger storage space to store M multiple hash metrics, where $M \geq 1$;

- To decrease the probability of false-positive, CCACK needs to run test algorithms M times, each of which with a different hash metrics;

Although CCACK can reduce the false-positive probability from $\frac{1}{2^8}$ to $(\frac{1}{2^8})^M$, it introduces both higher memory overhead and computation overhead. And when $M = 1$, the false-positive probability of CCACK is the same as NSB while having a more complex computation overhead. In fact, to overcome the collective-space problem in NC-based transmission, we only need a modified NSB ACK scheme (M-NSB) instead of the more complex CCACK.

We first elaborate how the original NSB ACK works. We denote the set of coding vectors received by node i to be B_i^r. When node i wants to broadcast about the feedback information of linear independent packets it currently has, it generates the feedback information as a vector z_i that satisfies:

$$z_i \cdot v = 0, \qquad \forall v \in B_i^r \tag{51}$$

Let V_i^r denote the linear space spanned by vectors in B_i^r. It is shown in [60] that:

Lemma 1 *With the above random construction of z_i, any vector $v' \in V_i^r$ must satisfy $z_i \cdot v' = 0$. And for any vector $v'' \notin V_i^r$, the probability of $z_i \cdot v'' = 0$ is $\frac{1}{2^8}$ when $GF(2^8)$ is used.*

The reason why NSB coded feedback may cause the collective-space problem is because NSB is not designed to convey the collective space of all downstream nodes but only the space relationship between the individual node pairs. To overcome this shortcoming while keeping the implementation at a low complexity level, we design the M-NSB coded feedback scheme. M-NSB has two different features from the original NSB:

1. Instead of generating z_i for set B_i^r, M-NSB generates z_i for set B_i^w, which is the coding vector set of all the re-encoded packets node i broadcasts. Then the condition z_i needs to satisfy becomes:

$$z_i \cdot v = 0, \qquad \forall v \in B_i^w \tag{52}$$

2. Node i stops broadcasting when there are L_i vectors in B_i^w are marked to be received by nodes in FS_i.

After a M-NSB ACK is generated, it is broadcast by the receiving node. M-NSB is different from CCACK in that M-NSB does not take nodes overhearing from different upstream nodes into account. This is for the objective of precisely measuring and controlling the total transmission cost for the whole network. In EENCR, each node has its own effective load and packets received by the same node but from different senders will be viewed as different traffic flows. By solving the collective-space problem for each sender separately, every coded packet can be effectively used for the decoding at the destination. Therefore, M-NSB addresses the second challenge in designing NC-based routing protocols.

Rate control

In EENCR, the routing engine component provides the effective load information, and the M-NSB component provides the receipt status of re-encoded packets to the forwarder set. We then design a rate control component to help each node decide when to start the broadcast and how fast it should broadcast.

We first give the following definition of traffic flow:

Definition 2 *A traffic flow f is defined as a 5-tuple $f = (S, T, x, j, i)$ to represent a load of packets originated at node S and destined at T with batch index x, which is forwarded from sender j to forwarder i.*

At each non-destination node i, EENCR maintains an array $B_i^v(f)$ to store linear independent packets received for each flow. We also define a binary active-flow indicator I_f for each flow (S, x, j, i). I_f is set to be false by default and is updated to true only when one of the following two conditions is satisfied:

1. Node i is the first member of FS_j for flow f;

2. Node i receives more than $K_i(f) - L_i(f)$ linear independent packets from node j for flow f, where $K_i(f)$ is the number of linear independent packets i expected to receive from j for flow and $L_i(f)$ is the effective load assignment of node i for flow f.

Every time there is a transmission opportunity for node i, one active-flow is chosen in a round-robin fashion. A re-encoded packet is generated by selecting non-zero elements in $GF(2^8)$ as re-encoding vectors for packets in $B_i^v(f)$. Node i then broadcasts this re-encoded packet and adding the re-encoding vectors into $B_i^w(f)$. Once the forwarder set of i has received $L_i(f)$ linear independent packets, I_f is set to false and the array for flow f will be flushed.

Performance evaluation

To characterize the feasibility and effectiveness of network coding in improving the energy efficiency , we experimentally evaluate the performance of EENCR in this section. We first present the experimentation methodology and then the measurement results.

Methodology

Testbed. We use the *NetEye* wireless sensor network testbed at Wayne State University [3]. Different from the environment of NetEye presented in Chapter 2, We have moved NetEye to a new location since 2011 due to university arrangements. Nonetheless, we did our best to keep the basic features of NetEye unchanged in the new location. In the new NetEye, we still deployed 130 TelosB motes, where every two closest neighboring motes are separated by 2 feet in an indoor environment. But the layout of the whole testbed is no longer a regular grid due to the constraints of the room.

Out of the 130 motes in NetEye, we randomly select 40 motes (with each mote being selected with equal probability) to form a random network for our experimentation. Each of these TelosB motes is equipped with a 3dB signal attenuator and a 2.45GHz monopole antenna.

In our measurement study, we set the radio transmission power to be -7dBm (i.e., power level 15 in TinyOS) such that multihop networks can be created. And we use the default MAC protocol provided in TinyOS 2.x.

Protocols studied. To understand the impact of network coding in improving the energy efficiency of wireless sensor networks, we comparatively study the following protocols:

- *EENCR*: the distributed NC-based routing protocol we proposed, which selects the optimal forwarder set for each node to minimize the transmission cost;
- *CTP*: a state-of-the-art collection tree protocol designed for data collection in sensor networks [27];
- *MORE*: the first NC-based opportunistic routing protocol that fully explores the routing diversity in the network by letting each forwarder to forward randomly coded packets;
- *CodeOR*: a NC-based opportunistic routing protocol that increases the concurrency of data flow by adding hop-by-hop ACK to the prototype of MORE.

We implement all four protocols in TinyOS 2.x. Due to the constraints of memory space of TelosB motes, which is only 10 kilobytes, and the short data payload length in sensor network applications, we choose a batch size of 8 for network coding operation instead of the mostly used batch size of 32 in wireless mesh networks.

Performance metrics. For each protocol we study, we evaluate their behavior based on the following metrics:

- *Delivery reliability*: percentage of information elements correctly received by the sink;
- *Delivery cost*: number of transmissions required for delivering an information element from its source to the sink;
- *Goodput*: number of valid information elements received by the sink per second;
- *Routing diversity*: number of forwarders selected to transmit a packet.

Different from the throughput metric used to evaluate the performance of NC-based routing protocols in [13] [50], in this study we use goodput instead. An information element is defined as **valid** if and only if it is linear independent to all elements that are in the same batch and received by the sink. And we do not study the routing diversity of CTP because its number of forwarders to transmit a packet is always one.

Traffic pattern. To experiment with both light and heavy traffic scenarios, we use two periodic data collection traffic patterns as follows:

- $S10$: out of all 40 nodes in the networks, 10 are selected as source nodes; Each source node periodically generates 40 information elements with an inter-element interval, denoted by Δ_r, uniformly distributed between 500ms and 3s; for EENCR, MORE and CodeOR, every consecutive 8 information elements compose a batch; this is to represent light traffic load scenarios.
- $S20$: same as $S10$ except that 20 nodes are selected as source nodes; this is to represent heavy traffic load scenarios.

Measurement results

In what follows, we first present the measurement results for light traffic pattern $S10$, then we discuss the case of heavy traffic pattern $S20$. In the figures of this section, we present the means and their 95% confidence intervals for the corresponding metrics.

Light data traffic

For the light traffic pattern $S10$, Figures 39 - 41 show the delivery reliability, delivery cost and goodput of different protocols. We found that EENCR and CTP provide high data delivery reliabilities (both are close to 100%) while MORE and CodeOR can only delivery 78% and 85% of the data to the sink on average. In the meantime, EENCR has a much lower delivery cost than CTP, i.e. a 26% reduction, in terms of average number of transmissions to deliver a packet but the delivery costs of MORE and CodeOR are around 400% and 300% of CTP respectively. Furthermore, EENCR enables a higher data goodput very close to the theoretical maximal value than all other three protocols.

The reasons for the inferior performance of MORE and CodeOR in our study are as follows:

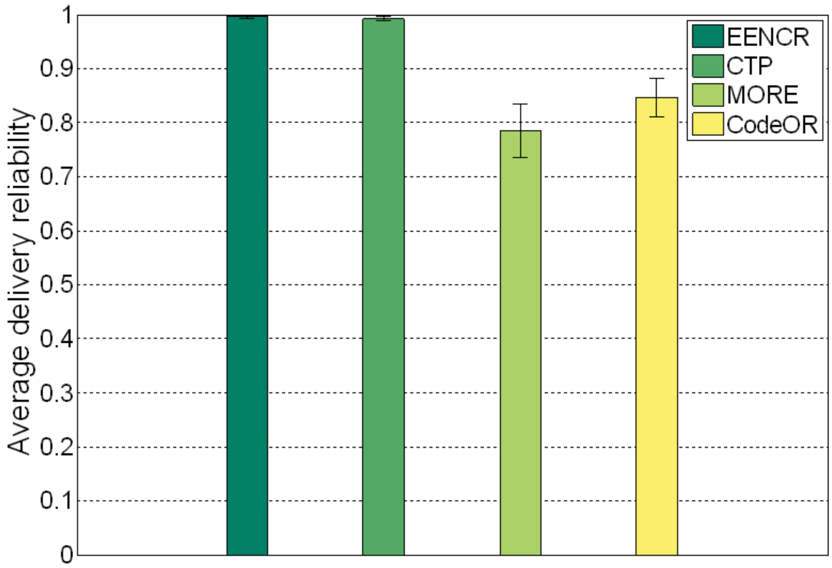

Figure 39: Delivery reliability: 10 sources

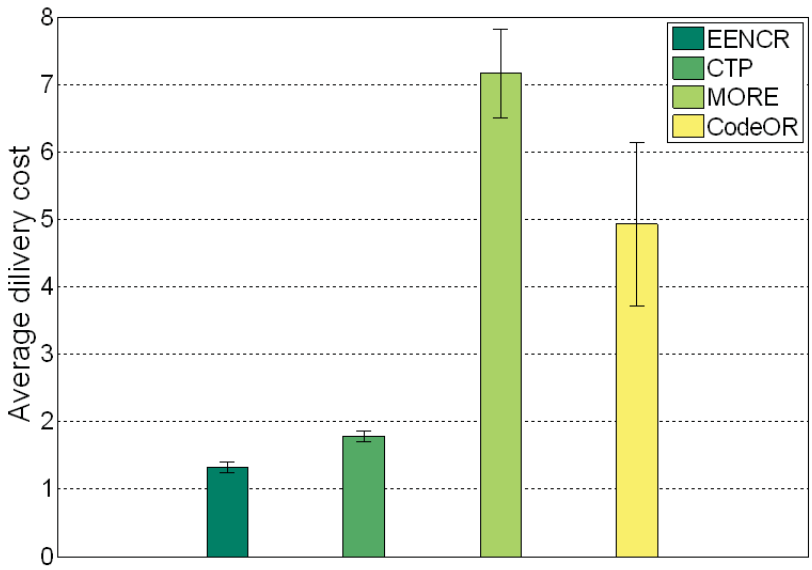

Figure 40: Delivery cost: 10 sources

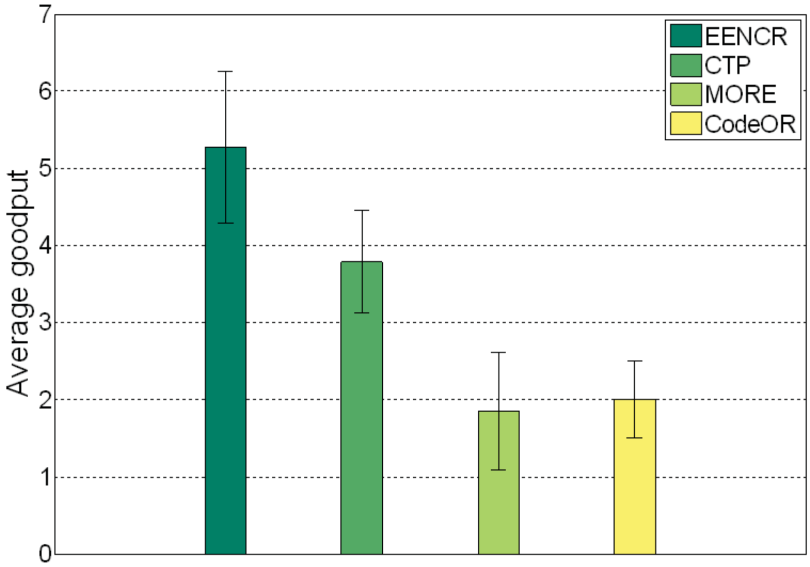

Figure 41: Goodput: 10 sources

1. The main design principle of MORE and CodeOR is to have all the forwarders encode and broadcast the packets they received. Although this principle made full use of the spatial routing diversity for wireless networks, having all nodes in a network would significantly increase the contention of the network and compromising its performance. On the other hand, EENCR adopts an optimal greedy approach that only allows forwarders that can contribute in reducing the total transmission cost to get involved in the forwarding process. This strategy also helps reduce the contention in the network, which further improves EENCR's performance.

2. Both MORE and CodeOR rely heavily on the assumption of a reliable end-to-end ACK scheme to make source nodes and intermediated nodes stop broadcasting after the destination received enough coded packets for a certain batch. However, end-to-end ACKs tend to be unreliable, and it takes non-negligible time for all the nodes in the network to get an end-to-end ACK for a certain batch from the destination.

To elaborate on the above observations, we compare the number of forwarders selected in EENCR, MORE and CodeOR and summarize the results in Figure 44. It is shown in this figure

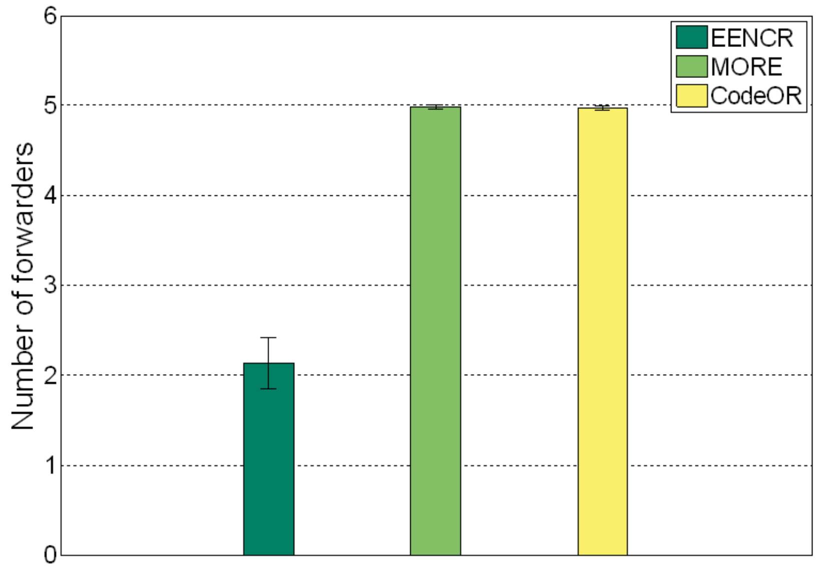

Figure 42: Routing diversity: 10 sources

that the average forwarders selected for each non-sink node in EENCR is around 2, but this number becomes 5 in MORE and CodeOR.

Heavy data traffic

To study the performance of EENCR in a more saturated network, we increase the number of sources to 20 to create a heavy traffic scenario $S20$, Figures 43 - 45 show the delivery reliability, delivery cost and goodput of different protocols. With heavier traffic in the network, EENCR is still able to provide a 98% data delivery reliability. Additionally, the reduction of EENCR compared to CTP has increased to 28%. This observation again is consistent with the design philosophy of EENCR. With heavier data traffic load in the network, the transmission cost of single path routing degrades. On the contrary, the transmission cost of EENCR still stays at a low level in that it fully explores and optimally leverages the wireless routing diversity in the network.

Meanwhile, the performance of MORE and CodeOR degrades even more severely than CTP due to similar reasons in the light traffic scenario. It is worthwhile to note that the goodput of CodeOR is even lower than MORE under $S20$. This is because CodeOR tries to increase the

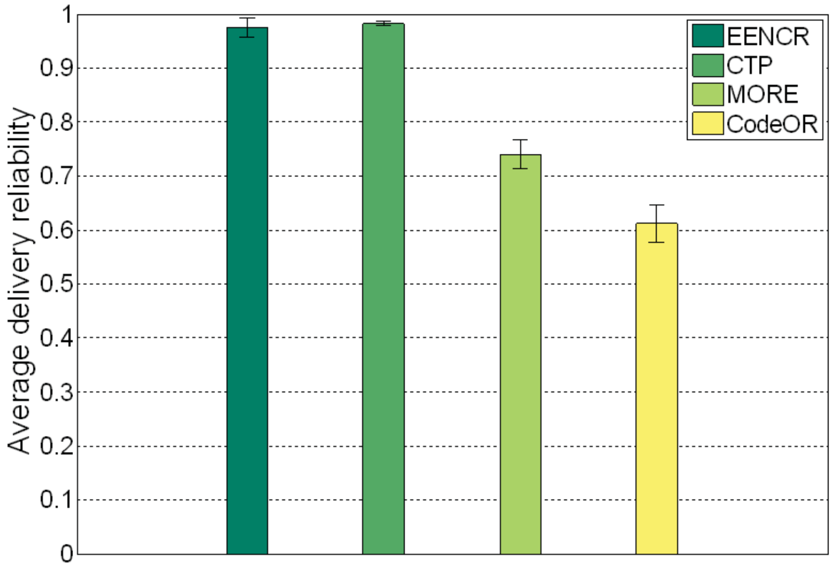

Figure 43: Delivery reliability: 20 sources

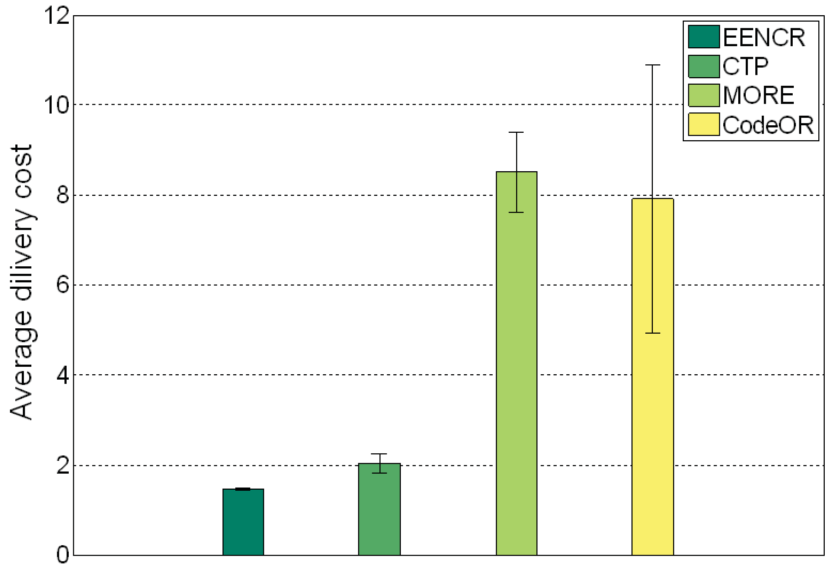

Figure 44: Delivery cost: 20 sources

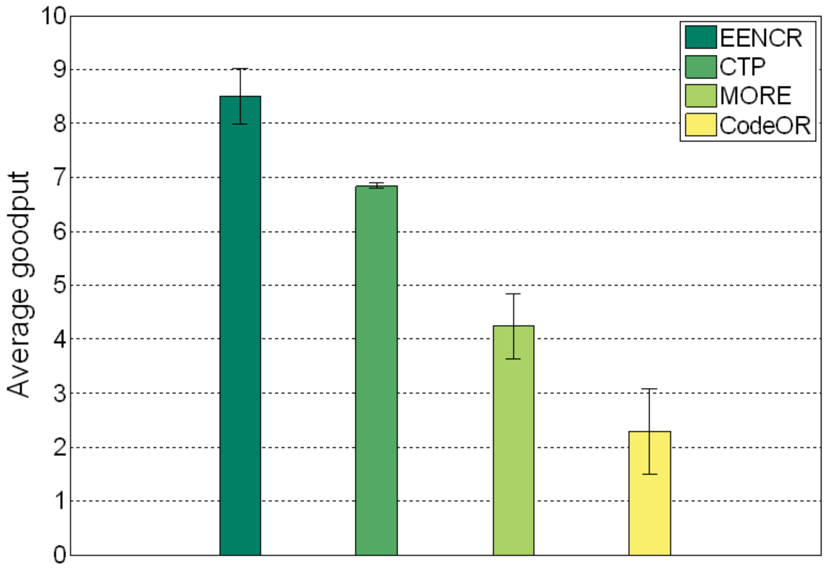

Figure 45: Goodput: 20 sources

concurrency of the network by allowing multiple flows for the same source to be injected in the network. However, it still has all the forwarders in the network to encode and forward packets towards the destination, which would result in high contention and poor delivery performance in the network. Injecting too many flows in the network without considering the negative effects brought by allowing every forwarder to perform forwarding operation can be disastrous in a network with heavy traffic, as shown in our experiment results. We show the routing diversity in terms of average number of forwarders selected in these NC-based protocols in Figure 46. This observation demonstrates, from another perspective, that it is of great importance and necessity to choose forwarder sets in NC-based routing protocols carefully.

Related work

Network coding was first proposed for wired networks in the pioneering paper [6]. By mixing packets at intermediate nodes during the transmission, the bandwidth can be saved and therefore the throughput of the whole network can be significantly improved. During the past years, network coding has been one of the most popular research topics in computer networks.

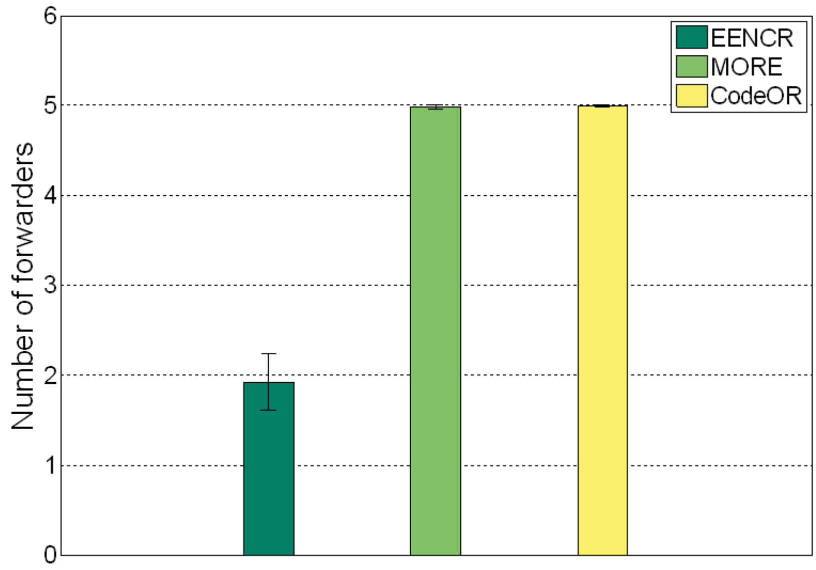

Figure 46: Routing diversity: 20 sources

Different coding schemes are designed, categorized into linear network coding and non-linear network coding. Compared with linear network coding, non-linear network coding has been reported to outperform linear coding in several studies [45] [16] [44] [17]. Especially in [17], it is shown that there are multi-source network coding problems for which non-linear coding has a general better performance on throughput. Nevertheless, according to the analysis from [47], linear network coding can provide a performance close to the best possible throughput while only requires a relative low complexity compared with the high complexity of non-linear coding.

Due to the broadcast nature in wireless communication, each intermediate node can receive redundant packets during the transmission in wireless networks. Network coding is one of the best choices to make use of these redundancies. By mixing redundant packets together and forwarding the mixed packet, the throughput of the wireless networks can be further improved. It is shown that linear coding functions can be designed randomly and independently at each node [31] [34]. Authors in these papers proposed a coding technique called random linear coding (RLC). Since RLC can be easily implemented in a distributed manner and it has a low complexity, it is widely used in wireless networks, including wireless sensor networks [28].

After network coding has been proved to be able to effectively use the overhearing redundancy in wireless environment, research on network coding in wireless networks has been following two different broad directions.

Network-coding-based multicast

Multicast has been well studied in wireless networks in the past few decades. Introducing network coding into multicast protocol, researchers find that the randomness of coded packets can effectively reduce the latency of multicast, therefore increase the network throughput.

Eryilmaz *et al.* [18] is the first work studying the delay performance gains from network coding. The authors study the problem on a wireless network model with one source and multiple receivers. Files are transferred from the source to receivers using network coding. The delay performance in this paper is defined as the average complete time of a file transmission. The authors study two different cases: 1) a file is broadcast to all receivers (broadcast case); 2) each receiver demands a different file (multiple unicast case). According to the theoretical analysis in this paper,there is a significant delay performance gain in both broadcast case and multiple unicast case via network coding, i.e., the average completion time is reduced.

Although network coding is proved to be able to provide average latency guarantee in [18], there is still a trade-off between the throughput and end-to-end latency for network coding in different wireless networks. Katabi *et al.* [23] used a simple example as follows to demonstrate this trade-off.

Suppose there are k packets needed to be sent from node A to B, link AB has a reliability of 50%. If node A sends these packets separately, it would require an expected number of transmission $4k$ including sending back k ACK packets. If all these packets are generated by A at the same time and therefore could be coded into k coded packets. Successfully sending these k coded packets would require an ETX of only $2k + 1$ including sending back only 1 ACK packet. If $k/2$ packets are generated first and has to be sent to B before the other $k/2$ packets are generated, these k packets could only be coded into two groups with $k/2$ coded packets each. The whole ETX for this transmission scheme is $2k + 2$ including sending back 2 ACK packets.

Zhang *et al.* [74] investigate the benefits of using Random Linear Coding (RLC) for unicast communications in a mobile Disruption Tolerant Network (DTN) under epidemic routing. In this paper, the authors propose the following coding and transmitting scheme: DTN nodes store and then forward random linear combinations of packets as they encounter other DTN nodes. The simulation results show that when there is one single file composed of several packets propagating in the network, when bandwidth is constrained, applying intra-flow RLC over packets can improve the delivery delay to deliver the whole file, and there is more improvement when the buffer in each node is limited. When there are multiple files propagating in the network, simulations results show that intra-flow RLC offers only slight improvement over the non-coded scheme when only bandwidth is constrained, but more significant improvement when both bandwidth and buffers are constrained.

The work in the above paragraph studies the benefits of network coding in DTN by a simulation based approach. Different from [74], Lin *et al.* [51] study this problem in a theoretical analysis framework. The theoretical analysis achieves similar conclusions as those in [74]. Based on the analysis, the authors also design a priority coding protocol, in which packets in the same file are divided into different groups with priorities and packets with higher priority would be coded and transmitted first. When the destination receives all coded packets for a certain level, it notifies the whole network and the source so that the same packets stored in the network will be dropped to further increase the performance of the network.

In both [74] and [51], the authors do not consider interferences in the network, which is reasonable only for sparse networks. Zhang *et al.* [73] conduct an analysis on the throughput-delay tradeoffs in mobile ad hoc networks (MANETs) with network coding, and compare results in the situation where only replication and forwarding are allowed in each node. The network model is built on both fast mobility model (i.i.d. mobility model) and slow mobility model (random walk model). The authors propose a k-hop relay scheme in a n -node MANET using RLC in MANETs and prove the trade-off between throughput and delay of the proposed scheme under two mobility models. Under fast mobility model, where $k = \Theta(\log n)$, the throughput

$T(n) = \Theta(1/n)$ and the average delay $D(n) = \Theta(\log n)$, where $T(n)$ represents throughput and $D(n)$ represents average delay. Under the slow mobility mode, where $k = \Theta(\sqrt{n})$, $T(n) = \Theta(1/n)$ and $D(n) = \Theta(\sqrt{n})$. This is the first work to study the trade-off between throughput and delay using RLC in MANETs. However, this study still uses the average delay as the metric instead of putting hard latency constraints on the analysis.

Katti *et al.* [39] propose COPE, a new architecture for wireless mesh networks. It is the first network coding that is implemented with the current network stack seamlessly. In the design of COPE, only inter-flow network coding is concerned. That means packets headed to the same next hop or generated by the same source cannot be encoded together under COPE. And COPE adopts an opportunistic coding scheme, which does not delay packets' transmissions for further coding opportunity. According to the theoretical analysis, not only can network coding bring a significant improvement on throughput, but also the MAC layer protocol can also improve the network throughput when it is combined with coding technique. COPE is implemented on a 20-node wireless network testbed. The experiment results show that COPE can increase the throughput of wireless mesh networks without modifying routing or higher layers.

Network-coding-based opportunistic routing

Other than network coding, opportunistic routing is another technique that fully explores the diversity of the broadcast nature in wireless communication. ExOR is the first opportunistic routing protocol and was proposed in [12]. Since then, extensive work has been conducted to further improve the forwarder candidate selection process in opportunistic routing. However, the essential component in opportunistic routing protocols incurs heavy communication cost of node coordination and requires a delicately designed MAC protocol.

As a continuous research of [12][39], Chachulski *et al.* [13] integrated intra-flow RLC and the opportunistic routing protocol in [12] to develop a new routing protocol called MORE in wireless mesh networks. The contribution of MORE is multi-dimensional. First, it makes use of the broadcast property of wireless communication to improve the network throughput without modifying the existing MAC layer, e.g., 802.11. Secondly, it adopts RLC for intra-flow network coding. RLC has a low complexity and is is easy to implement in a distributed system. Therefore, the network throughput is further improved. Thirdly, both the memory overhead and the header overhead are bounded within a reasonable range. MORE is also evaluated in a 20-node testbed and it outperform ExOR in both unicast and multicast traffic flow with a higher throughput.

Quite a few new protocols has been built based on MORE to further improve the throughput of NC-based opportunistic routing [50] [41] [28] [42] [75]. The basic idea of these studies is the natural combination of opportunistic routing and network coding because they both made use of the broadcast nature of wireless transmission. Koutsonikolas *et al.* [42] propose another intra-flow network coding architecture called Pacifier. Pacifier builds an efficient multicast tree and extends it to opportunistic overhearing. Then it applies intra-flow RLC technique to ensure the reliability. Both these two steps are similar with MORE. Besides these two components, Pacifier also applies a source rate control module to avoid the congestion in the network. Most importantly, Pacifier solves the ”crying baby” problem by having the source send batches of packets in a round-robin fashion. Not only large scale simulations but also a series of experiments in a 22-node wireless testbed show that Pacifier have a large improvement on average throughput compared with MORE. Similar to Pacifier, [28] proposed Rateless Deluge, the first implementation of NC-based opportunistic routing protocol in wireless sensor networks.

Zhu *et al.* [75] propose a hybrid coding scheme that does inter-flow coding first and intra-flow coding later. In the proposed scheme, packets are first encoded following the same coding scheme adopted by COPE. Then the encoded packets are divided into different batches. Encoded packets in the same batch are further encoded following the same coding scheme adopted by MORE. During the transmission, the whole system uses a multiple-path transmitting scheme to further improve the network throughput. The authors do a theoretical analysis on their proposed coding scheme in a simple wireless network model. Compared with COPE, the hybrid coding scheme has a significant improvement on both throughput and reliability in this network model. However, simulation or experiments are needed to further testify the efficiency of this hybrid scheme.

To further improve the throughput of wireless networks, Lin *et al.* [50] make use of hop-by-hop ACK and sliding window to allow different segments of packets to be transmitted in the network concurrently (CodeOR). However, it still adopts offline ETX metric to decide how many coded packets to transmit to ensure the end-to-end decodability. To be adaptive to the dynamic of wireless links, Koutsonikolas *et al.* [41] uses a Cumulative Coded ACK(CCACK) scheme to allow nodes to notifying their upstream nodes that they have received enough coded packets in a simple and low overhead way. The throughput of CCACK is shown to be 45% better than MORE. [41] is the most closely work related to our problem. The cumulative coded ACK scheme gives a good solution to the problem ”when should a sender stop broadcasting”. However, CCACK’s major objective is to minimize the broadcast cost at each sender/forwarder. This approach cannot give a global minimization on transmission cost for NC-based opportunistic routing. Furthermore, CCACK requires a high memory space and a relatively complex computation process, which is not suitable for resource-constrained sensing networks.

Concluding remarks

NC-based routing has drawn the interests of many researchers in wireless community. In this section we studied the minimal cost NC-based routing problem. We proposed the first effective load based mathematical framework to compute the cost of NC-based routing for a given topology. To the best of our knowledge, this is the first successful attempt towards measuring the energy consumption of NC-based routing. Our solution provides a formal theoretical method to measure the transmission cost of intra flow network coding routing protocols.

Based on this framework, we then studies the open problem of computing the optimal transmission cost of NC-based transmission and the corresponding routing braid. We were able to derive a distributed polynomial-time greedy algorithm for this problem an proved its optimality. We further studied the property of this algorithm and showed that the optimal routing braid does not necessarily contains the shortest single path route as expected in traditional routing and opportunistic routing protocols. Plus, we proved that the upper bound of the energy consumption for optimal routing braid is the same as that of single path routing in terms of expected number of transmissions.

Furthermore, we proposed EENCR, an energy-efficient NC-based routing protocol for resource-constrained sensor networks. In EENCR, we adopted the 4-bit link estimator [27] and our minimal cost forwarder set selection algorithm in the routing engine component. We then developed M-NSB, a coded feedback scheme without near-zero additional communication overhead and designed a rate control component to avoid the energy waste caused by unnecessary broadcast. EENCR incorporated the design philosophy of CTP [27], a state-of-art single path routing protocol in sensor networks, so that the complexity of protocol is maintained at a low level, which is of great importance and favorable on low-power distributed platforms, e.g., TelosB sensors. Experiment results of EENCR on the NetEye testbed showed that EENCR yields a high reliability as CTP, and has a transmission cost that is only around 72-75% of CTP. In the meantime, the goodput of EENCR is significantly improved from MORE and CodeOR because it adaptively selects the forwarders instead of utilizing the whole forwarder candidate set.

CHAPTER 4
PROACTIVE NETWORK CODING BASED PROTECTION

Preliminary

There has been a lot work done on protection against network failures in both wired and wireless networks. Existing protection techniques can be generally categorized into two classes: 1) proactive protection that sends the same data along two different paths simultaneously, which is also called 1+1 or 1+N protection, and 2) reactive protection that sends the data along one path at the beginning and switch to another path when there is a failure detected, which is also called 1:1 or 1:N protection. It is easy to see that both protection strategies have their own advantages and drawbacks. Proactive protection has zero response time when failures happen while having a higher transmission cost. Reactive protection has a lower transmission cost than proactive protection but requires failure detection mechanism and longer time to take actions.

Different from traditional wired networks, network failures in mission-critical wireless cyber-physical systems usually have the following characteristics:

1. Network failures in WCPS are usually transient (e.g., lower reliability in wireless transmission due to environment change), which means failed nodes and links can function normally after some time;

2. When transient failures happened in WCPS, it is usually not an efficient way to identify and replace the failed hardware because of both the transient nature of these failures and the extra high cost incurred by failure detection and correction operations.

Therefore, an important design principle in building a resilient mission-critical WCPS is to ensure efficient and fast data delivery in the presence of transient network failures by enabling proactive network protections. Making use of the broadcast nature of wireless communication, network coding has promising potentials in network protection because every coded packet contains the same amount of information entropy. Using network coding, every packet is basically equally useful when the destination retrieves the original information.

Recently, there has been some work on providing proactive protection using network coding in mesh networks [7] [37] [57]. However, most of the application scenarios for these work are in optical networks or require some specific routing structure to realize the protection scheme. Therefore, these work cannot be applied to the general scenarios of mission-critical cyber-physical systems. To cope with the requirement of reliable and real-time data delivery in mission-critical WCPS, we extend our solution to minimal cost NC- based routing in Chapter 3 to study the NC-based proactive protection problem in wireless sensor networks. The contribution of this study is as follows:

- We study the minimal cost 1+1 NC-based proactive protection problem. Different from the well-known minimal 2 node-disjoint path problem, we show that this new problem is NP-hard even in a simplified version through a reduction from the 2-partition problem. As a trivial note, we also point out and fix a mistake in the NP-hardness proof of the classic 2 integral network flow problem in [19].

- Motivated by the classic 2 node-disjoint path algorithm and Algorithm 3 we designed in Chapter 3, we propose a heuristic algorithm for the 1+1 NC-based proactive protection problem. This algorithm computes two node-disjoint braids that has a total transmission cost upper bounded by the 2 shortest node-disjoint paths.

- We further design and implement ProNCP, a proactive network coding based protection protocol, on TelosB sensor platforms. We evaluate the performance of ProNCP on our NetEye testbed by comparing it with a benchmark routing protocol (TNDP) that transmits data along 2 node-disjoint paths. Experiment results show that ProNCP performs better than TNDP in terms of reliability, transmission cost and goodput under both no-failure scenario and random transient failure scenarios.

The rest of this chapter is organized as follows: we first present the system model and problem definitions of this study. Then we study the complexity of 1+1 NC-based proactive protection problem and propose a heuristic algorithm. Based on this algorithm, we further implement ProNCP and evaluated its performance on the NetEye testbed. Before we conclude this chapter, we also discuss related work on proactive protection in wireless networks.

System model and problem definition

This study shares the similar system model and notations as in Chapter 3. We model a wireless network as a graph $G = (V, E)$ with node S as the source and T as the destination. For each node $i \in G$, we use U_i and D_i to denote the set of senders and receivers of i, respectively. And we denote the forwarder set of i as $FS_i \subset D_i$. For each link $i \to j \in E$, we denote ETX_{ij}^x as its expected number of transmission to deliver a packet with length x and $P_{ij}^x = \frac{1}{ETX_{ij}}$ as the corresponding link reliability. Since network coding will not change the packet length during the transmission, we use ETX_{ij} and P_{ij} for simplicity. Then we define $C_{iT}(x)$ as the transmission cost of delivering x linear independent packets from i to T, and $C_{iD_i}(x)$ as the expected number of broadcasts of node i when nodes in D_i collectively receive x linear independent coded packets from i. Assuming S needs to deliver K packets as a batch to T, we define K_i^j as the number of linear independent packets node i receives from node j.

Given a graph $G = (V, E)$ and K original packets to be delivered from S to T. We first define the 1+1 proactive protection problem with minimal transmission cost as follows:

Problem Q Given a graph $G = (V, E)$ with one source S and one destination T, find two node-disjoint NC-based routing braids B_1 and B_2 such that the total cost of delivering K linear independent packets to T along each braid is minimized.

The transmission objective of Problem **Q** is to deliver 2 copies of each piece of data generated by S to T, which is the same as the 2 node-disjoint path problem. However, the solution to the 2 node-disjoint path problem can only deal with single-node failures. On the contrary, the solution to Problem **Q** will be able to provide robust routing structure for sensor networks against up to F node failures, where $F = min(|V_{B_1}, V_{B_2}|) \geq 1$. Therefore, its solution can protect the network against random transient node failures.

1+1 NC-based proactive protection problem

In this section, we study the 1+1 network coding based proactive protection problem in detail. In traditional 1+1 protection schemes, the most common approach is to build 2 node-disjoint paths with the minimal total cost. This problem has been well studied and is solvable in polynomial time [65] [66] [11]. The basic idea of these algorithms is to make use of successive cycling cancellation methods in network flow theory. However, when network coding is introduced into wireless transmission, we will be able to further reduce the transmission cost for single data flow as we have proved in Chapter 3. Therefore, how to construct 2 node-disjoint routing braids with minimal total cost for NC-based transmission becomes an interesting and open problem. To propose the solution to this problem, we first explore its computation complexity.

Complexity study on problem $\mathbf{Q}$

Though constructing 2 node-disjoint paths with minimal cost for a single data flow can be solved efficiently for survivable networks. It is impossible to transplant the solution idea to construct 2 node-disjoint routing braids with minimal cost for NC-based transmission due to the following reasons:

- In NC-based transmission, the cost of the first hop broadcast does not follow the additive linear law as in traditional network flow theory;
- Routing braid has multiple paths at the second hop such that the traffic load on each path is dynamic depending on its order in the forwarder set instead of being static as in traditional network flow problems.

Towards better understanding the property of problem $\mathbf{Q}$, we study its computational complexity and propose the following theorem.

Theorem 12 *Problem* $\mathbf{Q}$ *is NP-hard.*

Proof To prove this theorem, we first look at Problem $\mathbf{Q}'$, a simpler version of Problem $\mathbf{Q}$ as follows:

Problem $\mathbf{Q}'$ The same as problem $\mathbf{Q}$ except that all the paths from S to T are node-disjoint to each other.

Since we are required to assign each non-terminal node to either braid B_1, braid B_2 or none of them. We are able to build a binary programming model for problem $\mathbf{Q}'$.

$$
\begin{aligned}
\text{Minimize: } C_1 + C_2 \quad = \quad & \frac{1}{1-\prod_{i=1}^{m}(1-x_i \cdot P_{2i-1})} \cdot \sum_{i=1}^{m} \frac{x_i \cdot P_{2i-1} \prod_{j=1}^{i-1}(1-x_j \cdot P_{2j-1})}{P_{2i}} \\
& + \frac{1}{1-\prod_{i=1}^{m}(1-y_i \cdot P_{2i-1})} \cdot \sum_{i=1}^{m} \frac{y_i \cdot P_{2i-1} \prod_{j=1}^{i-1}(1-y_j \cdot P_{2j-1})}{P_{2i}} \\
& + \max\{\frac{1}{1-\prod_{i=1}^{m}(1-x_i \cdot P_{2i-1})}, \frac{1}{1-\prod_{i=1}^{m}(1-y_i \cdot P_{2i-1})}\}
\end{aligned}
$$

such that

$$
\begin{aligned}
& x_i \in \{0,1\} \\
& y_i \in \{0,1\} \\
& x_i + y_i \leq 1 \\
& P_{2i} \geq P_{2(i+1)} \\
& 0 \leq P_{2i} \leq 1 \\
& 0 \leq P_{2i-1} \leq 1 \\
& \text{for } i = 1, 2, \ldots, m,
\end{aligned}
\tag{53}
$$

Although 0-1 programming is generally NP-hard, it does not necessarily result in the NP-hardness of this special class of 0-1 programming. To tackle this class of 0-1 programming, we propose the following lemma about the complexity of Problem $\mathbf{Q}'$:

Lemma 2 *Problem $\mathbf{Q}'$ is NP-hard.*

Proof We prove the NP-hardness of problem $\mathbf{Q}'$ via a reduction from the classic two-partition problem. There are different expressions of the 2-partition problem and we use the following optimization version:

Two-partition problem: Given a finite set A and a weight $w(a)$ for any element $a \in A$, partition set A into two subsets A_1 and A_2 such that the difference between $\sum_{a \in A_1} w(a)$ and $\sum_{b \in A_2} w(b)$ is minimized.

Without loss of generality, we assume that every element in the finite set of the two-partition problem has a positive weight. Given Y, an instance of the two-partition problem with set $X = \{X_1, X_2, \ldots, X_m\}$, we construct Z, an instance of problem $\mathbf{Q}'$ as follows. We first build a topology $S \rightarrow \{A_1, A_2, \ldots, A_m\} \rightarrow T$. For each $i = \{1, 2, \ldots, m\}$, we define $P_{SA_i} = 1 - 0.1^{w(X_i)}$ and $P_{A_iT} = 1$.

In this constructed instance of $\mathbf{Q}'$, it is straightforward to see that the objective function can be simplified to

$$C_1 + C_2 = 2 + \max\{\frac{1}{1 - \prod_{i=1}^{m}(1 - x_i P_{SA_i})}, \frac{1}{1 - \prod_{i=1}^{m}(1 - y_i P_{SA_i})}\} \tag{54}$$

To minimize Equation 54, the optimal solution must satisfy the following condition,

$$x_i + y_i = 1 \text{ for any } i \tag{55}$$

This means each node A_i must be either assigned to braid 1 or braid 2. This point can be proved through a simple contradiction. Suppose the optimal solution of Z has a node A_x not assigned braid 1 or braid 2. By assigning A_x to the braid that has a higher 1st hop broadcast cost, we can decrease this broadcast cost, which leads to a better solution to Z. Therefore, solving problem $\mathbf{Q}'$ is equivalent to solve the following problem:

$\mathbf{Q}'$ - Partition version: Partition set $\{A_1, A_2, \ldots, A_m\}$ into two subsets S_1 and S_2 such that the difference between $\prod^{A_i \in S_1} 1 - P_{SA_i}$ and $\prod^{A_j \in S_2} 1 - P_{SA_j}$ is minimized.

After a simple mathematical transformation, we can see that

$$\begin{aligned} \prod^{A_i \in S_1} 1 - P_{SA_i} &= 0.1^{\sum^{A_i \in S_1} w(X_i)} \\ \prod^{A_j \in S_2} 1 - P_{SA_j} &= 0.1^{\sum^{A_j \in S_2} w(X_j)} \end{aligned} \tag{56}$$

Through this equation it is readily to see that the partition version of $\mathbf{Z}$ is equivalent to Y, which means there is an optimal solution to $\mathbf{Z}$ if and only if there is an optimal solution to Y. From this we claim that there exists a one-to-one mapping from two-partition problem to Q'. Therefore problem Q' is NP-hard.

Having proved the NP-hardness of problem Q', the NP-hardness of problem Q is an immediate outcome.

Having proved Theorem 12, we show that it is impossible to develop a polynomial-time solution to even a simplified version of problem $\mathbf{Q}$. This finding motivates us to design an efficient heuristic algorithm to compute good solutions to problem $\mathbf{Q}$.

A finding in the NP-hardness proof for two-commodity integral flow problem

During our work in the complexity study on the problem of finding two node-disjoint routing braids with minimal cost, we find a technical mistake in the NP-hardness proof of two-commodity integral flow (TCIF) problem in the classic paper [19]. In this paper, the authors proposed a reduction from any instance of the satisfiability (SAT) problem to the TCIF problem. For any instance of A of the SAT problem, this paper denotes variables in A as $x_1, x_2, \ldots, x_n$ and the clauses in A as $C_1, C_2, \ldots, C_k$. For each variable x_i, p_i represents the number of positive occurrences of x_i and q_i represents the number of negative occurrences of x_i. A lobe L_i is then constructed for each x_i as shown in Figure 47. After connecting each lobe one by one and adding some extra nodes corresponding all the clauses in instance A. The authors proved that there exists an satisfiable assignment for A if and only if there exists two commodities of integral flow in the reduced instance of the TCIF problem.

However, this proof ignored the case when $p_i = 0$ or $q_i = 0$ for some x_i, which can affect the correctness of this proof. For example, if $p_i = 0$ for some x_i, the constructed lobe L_i has only the lower part. Under this case, when there is a satisfiable assignment which assigns $x_i = 0$ for the SAT instance A, the constructed TCIF instance cannot find two commodities of integral flow because each arc has only a capacity of 1 and lobe L_i cannot be used for two commodities of flow. Therefore, it is a lethal mistake for the whole proof.

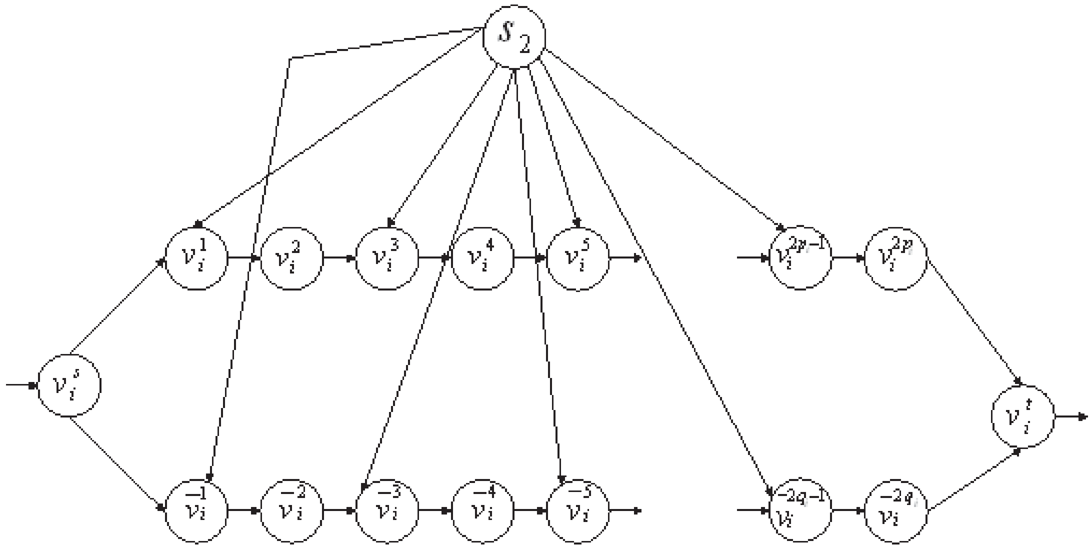

Figure 47: lobe i for variable x_i

Though this mistake invalids the whole correctness of this proof, we propose a simple patch to fix it:

EXPatch When $p_i = 0$ or $q_i = 0$ for variable x_i, we add a node v_i^{null} in the upper lobe or a node $\bar{v}_i^{null}$ in the lower lobe as in Figure 48.

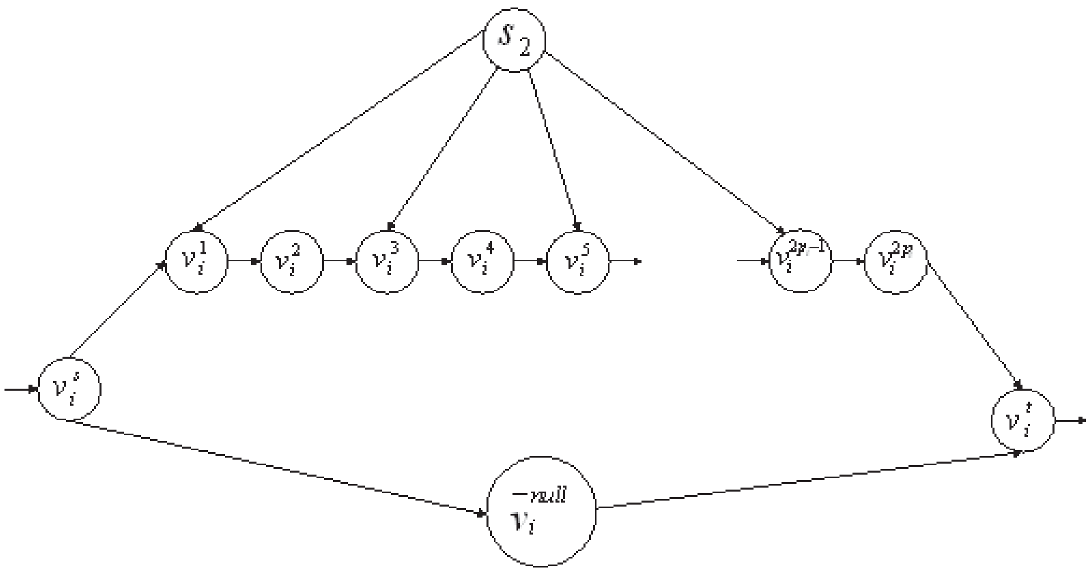

Figure 48: lobe i for variable x_i

Adding **EXPatch** into the NP-hardness proof of TCIF problem, it is readily to verify that the mistake in the original proof is now fixed because there are always the upper part and the lower part in each lobe. Note that this will not affect the proof of ”there exists an satisfiable assignment for an instance of SAT problem if and only if there exists two commodities of integral flow in the reduced instance of TCIF problem“ because there is no link from S_2 to node v_i^{null} or $\bar{v}_i^{null}$ for any i. Therefore, **EXPatch** fixes the mistake in [19] and completes the whole proof.

A heuristic algorithm for Problem Q

Since problem **Q** is NP-hard, in this section we propose a heuristic algorithm to this problem. This algorithm is motivated by both the classic algorithms for k node-disjoint paths with minimal cost [65] [66] [11], our effective load based mathematical framework for measuring the cost of NC-based transmission cost, and our optimal greedy single routing braid algorithm for network coding based routing in Chapter 3.

Algorithms proposed to construct k node-disjoint paths with minimal cost in a given directed graph [65] [66] [11] have a time complexity of $O(k|V|^3)$. In traditional protection studies, these algorithms have been showed to be effective in providing proactive protection to networks against single-node failures. However, by solving problem $\mathbf{Q_0}$ in Chapter 3, we find that the total transmission cost in wireless environment can be further reduced by fully exploring the routing diversity in sensor networks using NC-based routing because the minimal cost of NC-based routing is upper bounded by shortest single path routing in any DAG. Integrating solution ideas behind these two problems together, we propose a heuristic algorithm for problem **Q** that is able to find 2 node-disjoint braids with a total transmission cost upper bounded by two shortest node-disjoint paths and present it as Algorithm 4.

The first step of this heuristic algorithm is finding two node-disjoint paths with minimal total cost using the algorithm proposed in [65] as a reference point. We denote the two routing braids we want to construct as B_1 and B_2 and the two node-disjoint paths with minimal cost we find as $R_1 = S \to A_1^1 \to, \ldots, A_m^1 \to T$ and $R_2 = S \to A_1^2 \to, \ldots, A_n^2 \to T$. And we assign the initial of B_1 and B_2 as:

$$
\begin{aligned}
B_1 &= \{A_1^1, A_2^1, \ldots, A_m^1\} \\
B_2 &= \{A_1^2, A_2^2, \ldots, A_n^2\}
\end{aligned} \tag{57}
$$

Without loss of generalness, we assume that the cost of B_1 is larger than or equal to that of B_2, i.e., $C_{B_1} \geq C_{B_2}$. After the initialization of B_1 and B_2, we build an auxiliary graph G_1 by excluding all intermediate nodes in B_2 and all the links attached to these nodes from G. We then use Algorithm 3 to get the optimal single braid on G_1. Denoting the resulting braid as B_{single}^1, we update the first braid as:

$$B_1 = B_{single}^1 \tag{58}$$

With this new B_1, we then perform the same operations to update B_2. We build an auxiliary graph G_2 by excluding all intermediate nodes in B_2 and all the links attached to these nodes from G. Next we run Algorithm 3 again on G_2. Denoting the resulting braid as B_{single}^2, we will be able to update the second braid as:

$$B_2 = B_{single}^2 \tag{59}$$

After these operations, the algorithm stops and we will get two node-disjoint braids with a transmission cost upper bounded by two node-disjoint paths with minimal total cost. The rationale behind this heuristic approach is as follows:

- Instead of randomly dividing nodes into two braids or starting from two randomly paths, starting from two node-disjoint paths with minimal total costs can improve the efficiency of future node assignment process and guarantee the resulting braids have a total transmission cost upper bounded by the two shortest node-disjoint paths;

- Because transient failures are random in WCPS, we allow B_1 to have the priority to select nodes into the braid so that the cost of resulting braids can be balanced. With two node-disjoint braids of equal or balanced cost, the performance of WCPS, including transmission cost and throughput, can stay at a stable level under the existence of random transient failures. This feature is very desirable in modern mission-critical WCPS.

Algorithm 4 A heuristic algorithm for two node-disjoint braids construction

Input: a DAG $G = (V, E)$ with source S and destination T
Construct 2 minimal cost node-disjoint paths $\{R_1, R_2\}$ from S and T, where $C_{R_1} \geq C_{R_2}$
$B_1 = R_1, B_2 = R_2$
$G_1 = G$
for every node V_i in G_1 **do**
    if $V_i \in B_2$ **then**
        Remove V_i and all links attached to V_i from G_1
    end if
end for
Run Algorithm 3 on G_1 and denote the resulting braid as B^1_{single}
$B_1 = B^1_{single}$
$G_2 = G$
for every node V_i in G_2 **do**
    if $V_i \in B_1$ **then**
        Remove V_i and all links attached to V_i from G_2
    end if
end for
Run Algorithm 3 on G_2 and denote the resulting braid as B^2_{single}
$B_2 = B^2_{single}$
Stop and return $\{B_1, B_2\}$

Note: Different from Algorithm 3, we presented Algorithm 4 as a centralized algorithm. One reason we did this is because the construction of 2 minimal cost node-disjoint paths requires the complete information of the whole graph. The other reason, as we will show in the next section, is that a distributed version of Algorithm 4 would introduce large amounts of communication overhead to the network.

Protocol design and implementation

In the last section, we give a description on how to construct two node-disjoint routing braids with low transmission cost from a global perspective. In this section, we present the protocol design of 1+1 proactive NC-based protection (ProNCP) and details of its implementation.

ProNCP is essentially a NC-based routing protocol. It adopts most of EENCR's design principles we presented in Chapter 3, e.g., we implement the random network coding component, the coded feedback scheme and the rate control scheme the same as EENCR. However, we do not adopt the same distance-vector routing engine in EENCR. In EENCR, each node only needs to optimize its forwarder set without considering potential overlapping between the sub-braids of its forwarders and a distance-vector routing engine is sufficient for Algorithm 3. In ProNCP, on the contrary, to avoid braid overlapping is the most important constraint for braids construction. Therefore, a distance-vector routing engine is insufficient because a sender needs to know the whole graph of the network. A link-state routing component, on the other hand, will introduce high communication overhead and take up too much memory space, and is therefore inapplicable in resource-constrained mission-critical WCPS. To fill this gap, we conduct a long-time sampling test in our testbed to get packet delivery ratio for each link, perform offline computation of Algorithm 4 to get node-disjoint braids for each source, and assign these braids information into the implementation of ProNCP. We leave the design of a low-overhead distributed algorithm for two node-disjoint braids construction as a future research topic. Furthermore, we also add related control schemes in the packet forwarding component to make it fit ProNCP better.

Performance evaluation

To characterize the feasibility and effectiveness of network coding in providing proactive protection in mission-critical WCPS, we experimentally evaluate the performance of ProNCP in this section. We first present the experimentation methodology and then the measurement results.

Methodology

Testbed. We use the *NetEye* wireless sensor network testbed at Wayne State University [3]. The working environment of NetEye is different from that presented in Chapter 2, but the same as that presented in Chapter 3. 130 TelosB motes are deployed in an indoor environment, where every two closest neighboring motes are separated by 2 feet. The layout of the whole testbed is of a grid shape but with some slight variances due to the constraints of the room.

Out of the 130 motes in NetEye, we randomly select 60 motes (with each mote being selected with equal probability) to form a random network for our experimentation. Each of these TelosB motes is equipped with a 3dB signal attenuator and a 2.45GHz monopole antenna. In our measurement study, we set the radio transmission power to be -15dBm (i.e., power level 7 in TinyOS) such that multihop networks can be created. And we use the default MAC protocol provided in TinyOS 2.x.

Protocols studied. To the best of our knowledge, this is the first work to apply network coding against transient node failures in mission-critical WCPS. Some researchers have designed protocols to provide proactive protection using network coding in mesh networks [7] [37] [57]. However, these work cannot be applied to the general scenarios of mission-critical cyber-physical systems because they can only work under the existence of certain routing structures. Given the fact that most of works on routing selection for proactive protection in networks (wired and wireless) are based on the node-disjoint path construction algorithm, we study and compare the performance of the following protocols with the aim to understand the impact of network coding in improving the resilience of mission-critical WCPS against transient node failures,

- *ProNCP*: the 1+1 proactive NC-based protection protocol we propose in this chapter;
- *TNDP*: a routing protocol that sends data along two shortest node-disjoint paths to the receiver.

We implement both protocols in TinyOS 2.x. We choose a batch size of 8 for network coding operation as in Chapter 3. As we explained in the last section, we first conduct a long-time sampling test to get the packet delivery ratio of the whole network. Then we compute both node-disjoint paths and node-disjoint braids offline and assign the results into these two protocols. For TNDP protocol, we define the maximal number of retries for each packet to be 10 if no ACK of this packet was received by the sender/forwarder, this value is the same as what is used in CTP, a shortest single path routing protocol [27].

Performance metrics. For both protocols we study, we evaluate their behavior based on the following metrics:

- *Delivery reliability*: percentage of information elements correctly received by the sink;
- *Delivery cost*: number of transmissions required for delivering an information element from its source to the sink;
- *Goodput*: number of valid information elements received by the sink per second;

Different from the throughput metric used to evaluate the performance of NC-based routing protocols in [13] [50], in this study we use goodput instead. An information element is defined as **valid** if and only if it is linear independent to all packets that are in the same batch and received by the sink.

Topology. We randomly select 60 nodes out of 130 nodes in NetEye to form our experiment topology. From these 60 nodes, we randomly select 10 as source nodes. Each source node periodically generates 40 information elements with an inter-element interval, denoted by Δ_r, uniformly distributed between 500ms and 3s. For ProNCP, every consecutive 8 information elements compose a batch.

Transient node failure model

In our experiments, we deploy a periodic timer for all intermediate nodes in the network. Every time the timer at intermediate node V_i fires, V_i has a probability f to enter a transient failure status, i.e., not able to send or receive any packet. We comparatively study the performance of ProNCP and TNDP under different settings of f:

- $F0$: $f = 0$ for all intermediate nodes in the network; this is to represent the scenario where no node failure happens in the network.
- $F10$ $f = 0.1$ for all intermediate nodes in the network; this is to represent the scenario where intermediate nodes have a 10% chance to stop working for a short period of time.
- $F20$ $f = 0.2$ for all intermediate nodes in the network; this is to represent the scenario where intermediate nodes have a 20% chance to stop working for a short period of time.

Measurement Results

In what follows, we first present the measurement results for no failure scenario $F0$, then we discuss the case of failure pattern $F10$. In the figures of this section, we present the means and their 95% confidence intervals for the corresponding metrics.

No failure in the network

For the scenario that there is no failure in the network, we run ProNCP and TNDP 5 times each on the selected topology. Figures 49 - 51 show the delivery reliability, delivery cost and goodput of different protocols. In Figure 49, we find that both ProNCP and TNDP achieve a delivery reliability close to 100%. However, the average transmission cost of ProNCP is only 50% of that of TNDP, as shown in Figure 50. This observation is consistent with the design principle of Algorithm 4. By finding the optimal single braid on each auxiliary graph, we are able to significantly reduce the transmission cost of delivering two copies of data from sources to the root.

The reason why TNDP's transmission cost is much higher than ProNCP is because we set a maximal number of retries for each packet when the ACK of this packet is missing. We also try to set this maximal retries a smaller value, e.g. 5 and 8. But the corresponding reliability drops significantly to only 80%. On the contrary, we do not set any maximal number of retries in ProNCP. The number of coded transmissions for each received packet at any node is strictly assigned by the result of Algorithm 4. This further verifies the delivery efficiency of ProNCP over traditional node-disjoint paths algorithm.

In Figure 51, we find that the goodput of TNDP is slightly higher than ProNCP. This characteristic of ProNCP is acceptable. Different from EENCR, senders in ProNCP send two copies of each batch to the root. This proactive protection scheme doubles the traffic load in the whole network, making it more saturated. According to our experiment setting, the goodput of both ProNCP and TNDP are close to the capacity of the whole network.

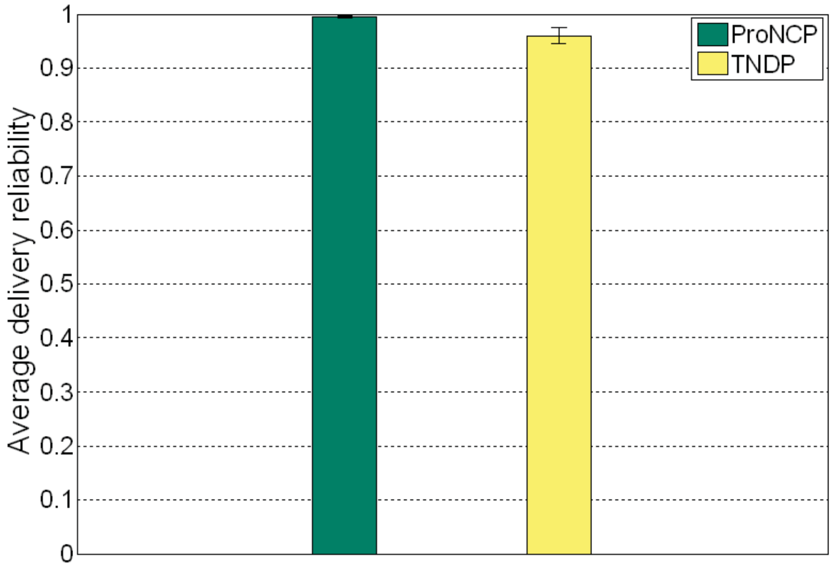

Figure 49: Delivery reliability: 10 sources without failure

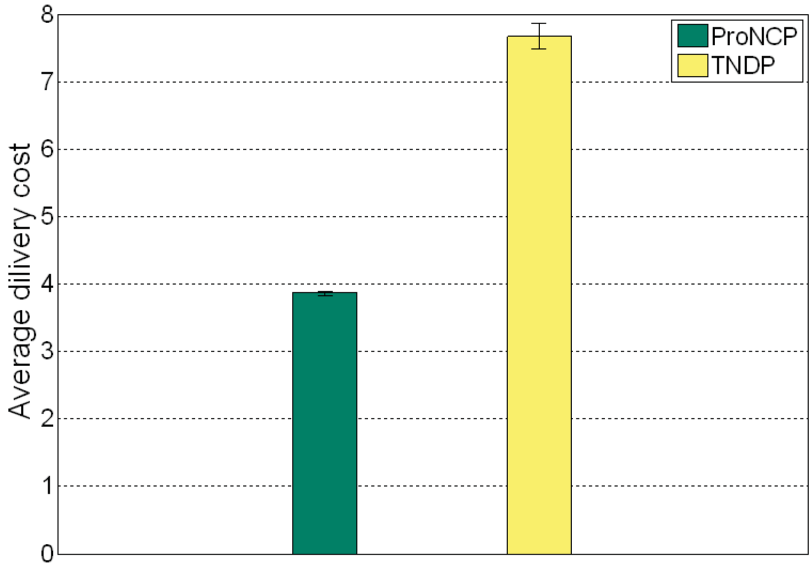

Figure 50: Delivery cost: 10 sources without failure

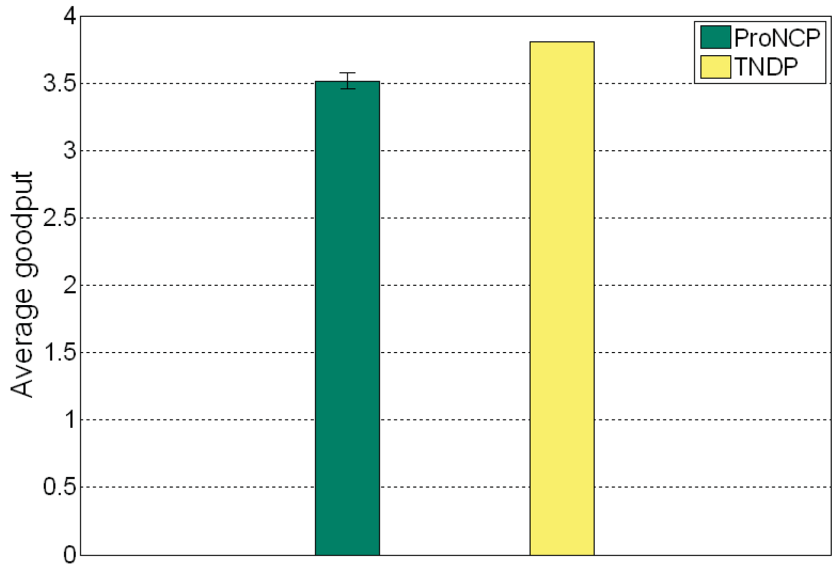

Figure 51: Goodput: 10 sources without failure

Random transient node failures in the network

After studying the performance of ProNCP under no failure scenario, we continue to evaluate the performance of ProNCP under the presence of random transient node failure. We run ProNCP and TNDP under each failure model for 10 times. Figures 52 - 54 show the performance of ProNCP and TNDP, including delivery reliability, delivery cost and goodput under both failure models. It is observed in Figure 52 that ProNCP is able to keep the delivery reliability close to 100% under both $F10$ and $F20$ failure models. On the contrary, The delivery reliability of TNDP degrades to 91% under $F10$ model and drops to 80% under $F20$ model. This figure proves that ProNCP is able to provide resilient against transient node failures for mission-critical WCPS.

Figure 53 shows that even under the existence of transient node failures, the average transmission cost of ProNCP is kept stable at a very low level. Comparatively, the average transmission cost of $TNDP$ slightly increases in $F10$ case, and drastically increases by 30% while still not able to guarantee data delivery in both failure models. This huge increase of transmission cost in TNDP is because we set the maximal number of retransmissions to be 10 for

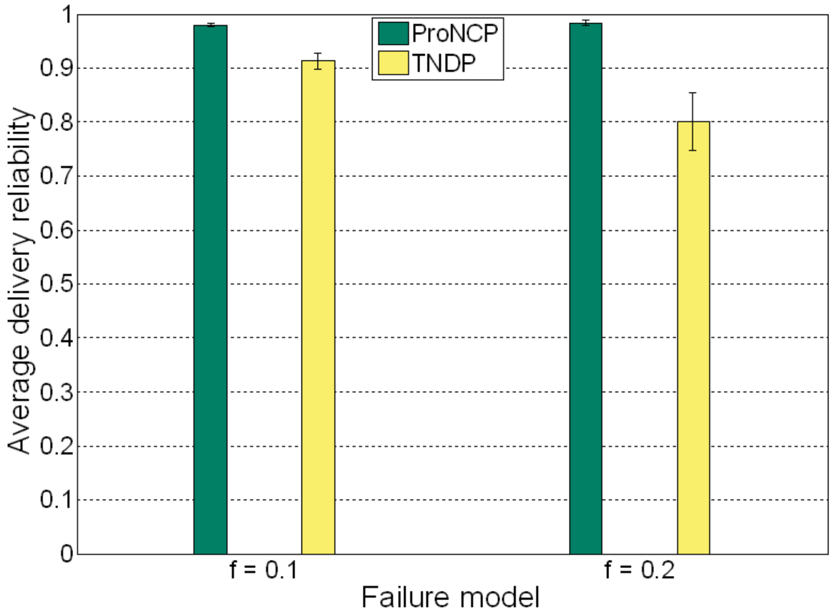

Figure 52: Delivery reliability: 10 sources with failures

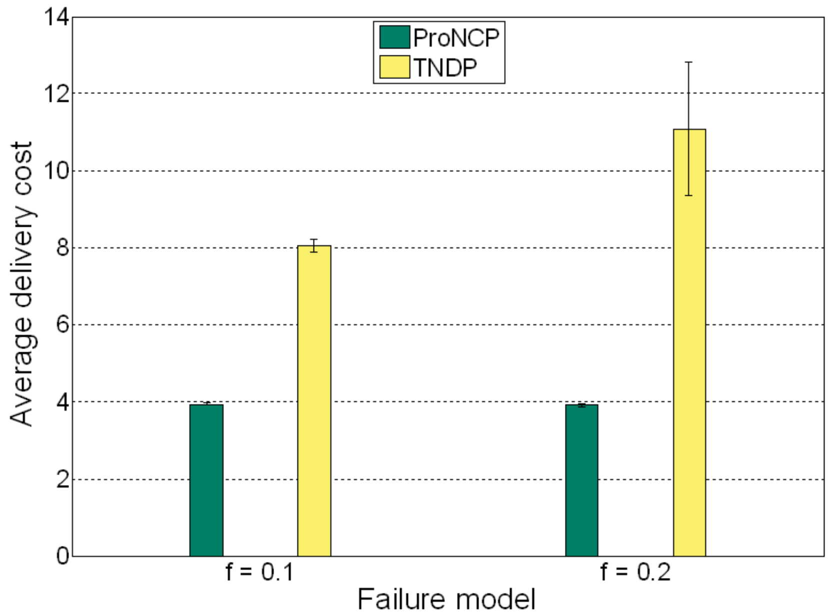

Figure 53: Delivery cost: 10 sources with failures

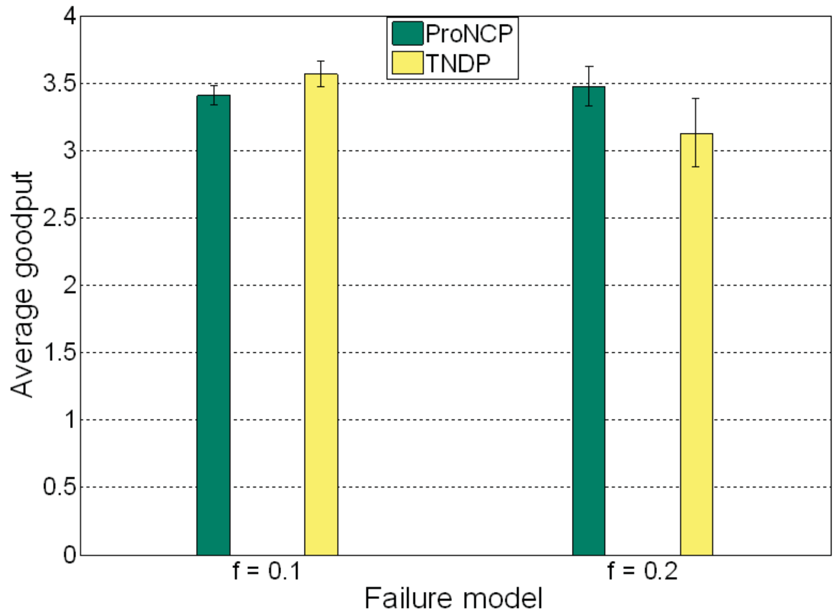

Figure 54: Goodput: 10 sources with failures

each packet. Under $F0$ scenario where no transient node failure happened, usually a packet is successfully transmitted over a link before the maximal number of retransmissions is reached. When intermediate nodes randomly enter transient failure status, under which they cannot receive or send packet, other working nodes have to retransmit packets for more times. The higher transient failure probability is, the higher the probability that a node has to keep retransmitting a packet till reaching maximal retries will be. On the contrary, the transmission cost of ProNCP is about the same in both $F10$ and $F20$ compared to the average number of transmissions in $F0$ scenario. This observation proves again the necessity and importance of an optimal algorithm for forwarder set selection in NC-based routing protocols. And it also shows that keep retransmitting under transient node failure cannot bring extra guarantee on reliability but only increase the transmission cost.

Furthermore, the difference between ProNCP's goodput and TNDP's goodput is very little under $F10$ model. And the goodput of ProNCP is even higher than that of TNDP in $F20$. This observation also demonstrates that ProNCP is capable of guaranteeing high data delivery and goodput under various transient node failures.

As a summary, in this section we show that ProNCP is resilient against the dynamics of wireless environment, i.e., transient node failures, in mission-critical WCPS. It is able to provide 1+1 proactive protection to the network with a significant lower transmission cost than the class proactive protection protocol, and maintain a high delivery reliability and goodput under different random transient node failure models.

Related work

There has been a lot work done on protection against node/link failure in both wired and wireless networks. Most existing protection techniques can be categorized into two classes: 1) proactive protection that sends the same data along two different paths simultaneously, which is also called 1+1 protection. 2) reactive protection that sends the data along one path at the beginning and switch to another path when there is a failure detected, which is also called 1:1 protection. It is straightforward to see that reactive protection has a lower transmission cost than proactive protection while proactive protection needs no response time or failure detection mechanism when failures happened in the network.

In proactive protection, many work focus on constructing node/link disjoint paths such that any single node/link failure will not affect the delivery of data to the destination. Several papers [65] [66] [11] studied disjoint paths in a network and proposed an algorithm to compute k minimum weight node-disjoint paths with a complexity of $O(kN^2)$ where N is the number of nodes in the network. Based on this result, many works have been done. Srinivas *et al.* [64] proposed an algorithm with a complexity of $O(kN^3)$ that controls the transmission power of the source node and compute the corresponding k node-disjoint paths with minimum energy cost in wireless networks. The wireless broadcast nature was considered in this paper for calculating the minimum energy consumption.

Recently, there has been some research on providing protection using network coding. Al-Kofahi *et al.* [7] enhanced the survivability of the information flow between two communicating nodes S and T without compromising the maximum achievable $S-T$ information rate. The authors claimed that most of the links in a network are not bottleneck links, which means that link failures are more likely to affect non-bottleneck links than links in the min-cut. Therefore, they can enhance the survivability of the $S - T$ information flow without reducing the useful $S - T$ rate below the max-flow, if protection is provided to the non-bottleneck links only. The system model of this work is in wired network and the solution cannot provide complete proactive protection to the network.

Kamal *et al.* [37] [57] studied the 1+N protection in the optical network against single link failure. By sending network coded packets on the protection Steiner tree in parallel with the working traffic, the proposed 1+N protocol is able to recover from any single link failure without enduring the delay from switching to the backup path. This problem is strongly NP-hard. And the heuristic solutions proposed in these two papers requires specific routing structure to ensure the protection, which is not realistic in wireless environment.

Braided multipath routing was first proposed in [25]. The major goal of braided multipath routing is to provide reactive protection in networks. After a single path is calculated as the main path, each non-destination node selects another path from itself to the destination. In this way, the data flow can always be switched to another path when there is a failure on the main path. Braided multipath routing can significantly improve the reliability of the network by having a higher connectivity than single path routing [56]. However, it cannot be applied into traditional proactive protection due to high transmission cost.

From the discussion above, we can see that traditional 1+1 protection in wireless network has a low throughput since it does not fully explore the broadcast nature of wireless transmission. Furthermore, packets received by the destination with the same packet number make the transmission redundant, which will increase the transmission cost.

On the contrary, protocols using network coding with opportunistic forwarding [13] [50] [41] have a higher throughput than regular single path routing because any packet received by the destination is not redundant as long as it is linear independent with packets already received by the destination. In the meantime, no node coordination is required between nodes within the same forwarder candidate set. Additionally, network coding with opportunistic forwarding has some implicit proactive protection scheme because the destination can decode all K original packets in the batch as long as it receive any K linear independent packets of this batch.

However, this type of protocols may have high transmission cost caused by no node coordination cost. Furthermore, even though network coding protocols have some implicit proactive protection scheme, they cannot guarantee full proactive protection, i.e., there are cases that one single node failure will lead the destination not receiving K linear independent packets unless it sends retransmission request to the source node.

Having seen both the benefit (the higher throughput and the implicit proactive protection) and drawbacks (high transmission cost and partial protection) brought by wireless network coding, we are motivated to design a network coding protocol for wireless networks in this chapter, such that it can provide full proactive protection against random transient node failures while keeping the high throughput by exploring the broadcast nature of wireless transmission with a low transmission cost.

In [13][50][41], protocols chose nodes with lower delivery cost to the destination into forwarder candidate set. This forwarder selection methods can increase network throughput but increase transmission cost as well because it was originally designed for opportunistic routing. In opportunistic routing, forwarders of the same node are prioritized. A forwarder can only forward the packet it received when no forwarders with higher priority successfully forwarded the packet. In this fashion, network transmission cost can be controlled at a low level. However, in network coding based opportunistic forwarding protocols, every forwarder can forward coded packets when the MAC is ready[13]. This approach did increase the network throughput with no need to design any specific MAC protocol. But if we still adopt the forwarder selection methods designed for opportunistic routing, the transmission cost will be increased.

Concluding remarks

NC-based routing has drawn the interests of many researchers in wireless community. Particularly, researchers have been trying to apply this technique into proactive protection for networks. In this section we study how to design energy-efficient network coding based solution in mission-critical wireless cyber-physical systems. Specifically, we study how to provide 1+1 proactive protection in sensor networks. We formally defined the two node-disjoint routing braids problem and prove its NP-hardness via a reduction from 2-partition problem. We then design a heuristic node assignment algorithm to compute two node-disjoint braids with a lower transmission cost than any two node-disjoint paths in the network. Based on this algorithm, we propose ProNCP, a proactive NC-based protection protocol. ProNCP inherits similar modules and components in EENCR, but we add corresponding control schemes to make the implementation satisfy the requirement of proactive protection in mission-critical WCPS.

We evaluate the performance of ProNCP on the NetEye testbed by comparing it with the two shortest node-disjoint paths algorithm (TNDP), the most classic approach in proactive protection. When there is no failure happening in the network, ProNCP is able to achieve a delivery reliability close to 100% with only half of the cost of of TNDP. When intermediate nodes have a probably of randomly entering transient failure state, the delivery reliability of TNDP degrades significantly while ProNCP is still able to maintain a high reliability and a low transmission cost. The resilience of ProNCP shown in the evaluation demonstrates the benefits of network coding in providing proactive protection for mission-critical WCPS. Future work towards this research direction includes the design of a distributed node-disjoint braids construction algorithm with low communication overhead.

CHAPTER 5
CONCLUSION

In this dissertation, we have studied two classic in-network processing methods, packet packing and network coding, as well as their performance in mission-critical wireless networked sensing and control. Through comprehensive theoretical study, we first demonstrate that both techniques can significantly improve network performance in providing real-time, efficient and resilient services to mission-critical WCPS. Based on these findings, we designed and implemented:

1. *tPack*, a utility-based packet packing scheduling protocol;
2. *EENCR*, an optimal minimal cost NC-based routing protocol;
3. *ProNCP*, a heuristic two node-disjoint braids proactive NC-based protection protocol.

Through extensive performance evaluation in our NetEye testbed, we showed that these three protocols outperform other in-network processing protocols, including some state-of-the-art protocols. Both our theoretical and experimental results shown in this dissertation provide deep insights on in-network processing protocol design for mission-critical wireless networked sensing and control systems.

To summarize, we studied in-network processing in wireless CPS from a real-time, efficiency and resilience perspective in this dissertation. And our findings will shed lights for future research in mission-critical wireless networked sensing and control system.

REFERENCES

[1] An event traffic trace for sensor networks. http://www.cs.wayne.edu/~hzhang/group/publications/Lites-trace.txt.

[2] Iowa's timesync component. http://tinyos.cvs.sourceforge. net/viewvc/tinyos/tinyos-2.x-contrib/iowa/T2.tsync/.

[3] NetEye testbed. http://neteye.cs.wayne.edu/neteye/home.php.

[4] IETF 6LowPAN working group. http://www.ietf.org/ html.charters/6lowpan-charter.html.

[5] TinyOS. http://www.tinyos.net/.

[6] R. Ahlswede, Ning Cai, S.-Y.R. Li, and R.W. Yeung. Network information flow. *Information Theory, IEEE Transactions on*, 46(4):1204 –1216, July 2000.

[7] O.M. Al-Kofahi and A.E. Kamal. Max-flow protection using network coding. In *Communications (ICC), 2011 IEEE International Conference on*, pages 1–5, 2011.

[8] A. Arora, R. Ramnath, E. Ertin, P. Sinha, S. Bapat, V. Naik, V. Kulathumani, Hongwei Zhang, H. Cao, M. Sridhara, S. Kumar, N. Seddon, C. Anderson, T. Herman, N. Trivedi, C. Zhang, M. Gouda, Y. R. Choi, M. Nesterenko, R. Shah, S. Kulkarni, M. Aramugam, L. Wang, D. Culler, P. Dutta, C. Sharp, G. Tolle, M. Grimmer, B. Ferriera, and K. Parker. Exscal: Elements of an extrem scale wireless sensor network. In *IEEE RTCSA*, 2005.

[9] Anish Arora, Prabal Dutta, Sandip Bapat, Vinod Kulathumani, and Hongwei Zhang et al. (17 authors). A line in the sand: A wireless sensor network for target detection, classification, and tracking. *Computer Networks (Elsevier)*, 46(5), 2004.

[10] Luca Becchetti, Peter Korteweg, Alberto Marchetti-Spaccamela, Martin Skuttella, Leen Stougie, and Andrea Vitaletti. Latency constrained aggregation in sensor networks. In *European Symposium on Algorithms (ESA)*, 2006.

[11] R. Bhandari. Optimal physical diversity algorithms and survivable networks. In *Computers and Communications, 1997. Proceedings., Second IEEE Symposium on*, pages 433 –441, jul 1997.

[12] Sanjit Biswas and Robert Morris. ExOR: Opportunistic multi-hop routing for wireless networks. In *ACM SIGCOMM*, 2005.

[13] Szymon Chachulski, Michael Jennings, Sachin Katti, and Dina Katabi. Trading structure for randomness in wireless opportunistic routing. In *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '07, pages 169–180, New York, NY, USA, 2007. ACM.

[14] Amol Deshpande, Carlos Guestrin, Wei Hong, and Samuel Madden. Exploiting correlated attributes in acquisitional query processing. Technical report, Intel Research - Berkeley, 2004.

[15] Amol Deshpande, Carlos Guestrin, Samuel Madden, Joseph Hellerstein, and Wei Hong. Model-driven data acquisition in sensor networks. In *VLDB*, 2004.

[16] R. Dougherty, C. Freiling, and K. Zeger. Linearity and solvability in multicast networks. *Information Theory, IEEE Transactions on*, 50(10):2243–2256, Oct. 2004.

[17] R. Dougherty, C. Freiling, and K. Zeger. Insufficiency of linear coding in network information flow. *Information Theory, IEEE Transactions on*, 51(8):2745–2759, Aug. 2005.

[18] A. Eryilmaz, A. Ozdaglar, and M. Medard. On delay performance gains from network coding. In *Information Sciences and Systems, 2006 40th Annual Conference on*, pages 864 –870, 2006.

[19] S. Even, A. Itai, and A. Shamir. On the complexity of time table and multi-commodity flow problems. In *Proceedings of the 16th Annual Symposium on Foundations of Computer Science*, pages 184–193, Washington, DC, USA, 1975. IEEE Computer Society.

[20] Kai-Wei Fan, Sha Liu, and Prasun Sinha. Scalable data aggregation for dynamic events in sensor networks. In *ACM SenSys*, 2006.

[21] Qing Fang, Feng Zhao, and Leonidas Guibas. Lightweight sensing and communication protocols for target enumeration and aggregation. In *ACM MobiHoc*, 2003.

[22] Gerd Finke, Vincent Jost, Maurice Queyranne, and Andras Sebo. Batch processing with interval graph compatibilities between tasks. *Discrete Applied Mathematics (Elsevier)*, 156, 2008.

[23] Christina Fragouli, Dina Katabi, Athina Markopoulou, Muriel Medard, and Hariharan Rahul. Wireless network coding: Opportunities & challenges. In *Military Communications Conference, 2007. MILCOM 2007. IEEE*, pages 1–8, Oct. 2007.

[24] H.N. Gabow. An efficient implementation of edmonds' algorithm for maximum matchings on graphs. *Journal of ACM*, 23:221–234, 1975.

[25] Deepak Ganesan, Ramesh Govindan, Scott Shenker, and Deborah Estrin. Highly-resilient, energy-efficient multipath routing in wireless sensor networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 5(4):11–25, October 2001.

[26] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of* NP-*Completeness*. Freeman and Company, 1979.

[27] Omprakash Gnawali, Rodrigo Fonseca, Kyle Jamieson, David Moss, and Philip Levis. Collection tree protocol. In *ACM SenSys*, 2009.

[28] Andrew Hagedorn, David Starobinski, and Ari Trachtenberg. Rateless deluge: Over-the-air programming of wireless sensor networks using random linear codes. In *Proceedings of the 7th international conference on Information processing in sensor networks*, IPSN '08, pages 457–466, Washington, DC, USA, 2008. IEEE Computer Society.

[29] Tian He, Brian M. Blum, John A. Stankovic, and Tarek Abdelzaher. AIDA: Adaptive application independent data aggregation in wireless sensor networks. *ACM Transaction on Embedded Computing System*, May, 2004.

[30] Joseph Hellerstein, Wei Hong, Samuel Madden, and Kyle Stanek. Beyond average: Toward sophisticated sensing with queries. In *IPSN*, 2003.

[31] T. Ho, R. Koetter, M. Medard, D.R. Karger, and M. Effros. The benefits of coding over routing in a randomized setting. In *Information Theory, 2003. Proceedings. IEEE International Symposium on*, pages 442–, June-4 July 2003.

[32] Dorit S. Hochbaum. *Approximation Algorithms for* NP*-hard Problems*. PWS Publishing Company, 1997.

[33] IETF. Routing over low power and lossy networks (ROLL) working group. http://www.ietf.org/html.charters/roll-charter.html.

[34] S. Jaggi, P.A. Chou, and K. Jain. Low complexity algebraic multicast network codes. In *Information Theory, 2003. Proceedings. IEEE International Symposium on*, pages 368–368, June-4 July 2003.

[35] Ashish Jain, Marco Gruteser, Mike Neufeld, and Dirk Grunwald. Benefits of packet aggregation in ad-hoc wireless network. Technical Report CU-CS-960-03, University of Colorado at Boulder, 2003.

[36] Raj Jain. *The Art of Computer Systems Performance Analysis*. John Wiley & Sons, Inc., 1991.

[37] Ahmed E. Kamal. 1 + n network protection for mesh networks: network coding-based protection using p-cycles. *IEEE/ACM Trans. Netw.*, 18(1):67–80, February 2010.

[38] Anna R. Karlin, Claire Kenyon, and Dana Randall. Dynamic TCP acknowledgement and other stories about $\frac{e}{e-1}$. In *ACM STOC*, 2001.

[39] Sachin Katti, Hariharan Rahul, Wenjun Hu, Dina Katabi, Muriel Médard, and Jon Crowcroft. Xors in the air: practical wireless network coding. *IEEE/ACM Trans. Netw.*, 16(3):497–510, 2008.

[40] Dzmitry Kliazovich and Fabrizio Granelli. Packet concatenation at the ip level for performance enhancement in wireless local area networks. *Wireless Networks*, 14:519–529, 2008.

[41] D. Koutsonikolas, Chih-Chun Wang, and Y.C. Hu. Ccack: Efficient network coding based opportunistic routing through cumulative coded acknowledgments. In *INFOCOM, 2010 Proceedings IEEE*, pages 1 –9, 2010.

[42] Dimitrios Koutsonikolas, Y. Charlie Hu, and Chih-Chun Wang. Pacifier: High-throughput, reliable multicast without.

[43] Rajnish Kumar, Matthew Wolenetz, Bikash Agarwalla, JunSuk Shin, Phillip Hutto, Arnab Paul, and Umakishore Ramachandran. DFuse: A framework for distributed data fusion. In *ACM SenSys*, 2003.

[44] April Rasala Lehman. *Network coding*. PhD thesis, Cambridge, MA, USA, 2005. Supervisor-Sudan, Madhu.

[45] April Rasala Lehman and Eric Lehman. Complexity classification of network information flow problems. In *SODA '04: Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 142–150, Philadelphia, PA, USA, 2004. Society for Industrial and Applied Mathematics.

[46] Ming Li, Hua Zhu, Yang Xiao, Imrich Chlamtac, and B. Prabhakaran. Adaptive frame concatenation mechanisms for qos in multi-rate wireless ad hoc networks. In *IEEE INFOCOM*, 2008.

[47] S.-Y.R. Li, R.W. Yeung, and Ning Cai. Linear network coding. *Information Theory, IEEE Transactions on*, 49(2):371–381, Feb. 2003.

[48] Tianji Li, Qiang Ni, David Malone, Douglas Leith, Yang Xiao, and Thierry Turletti. Aggregation with fragment retransmission for very high-speed WLANs. *IEEE/ACM Transactions on Networking*, 17(2):591–604, 2009.

[49] Xin Li, Young Jin Kim, Ramesh Govindan, and Wei Hong. Multi-dimensional range queries in sensor networks. In *ACM SenSys*, 2003.

[50] Yunfeng Lin, Baochun Li, and Ben Liang. Codeor: Opportunistic routing in wireless mesh networks with segmented network coding. In *Network Protocols, 2008. ICNP 2008. IEEE International Conference on*, pages 13 –22, 2008.

[51] Yunfeng Lin, Baochun Li, and Ben Liang. Stochastic analysis of network coding in epidemic routing. *Selected Areas in Communications, IEEE Journal on*, 26(5):794–808, June 2008.

[52] Junning Liu, Micah Adler, Don Towsley, and Chun Zhang. On optimal communication cost for gathering correlated data through wireless sensor networks. In *ACM MobiCom*, 2006.

[53] Kejie Lu, Dapeng Wu, Yi Qian, Yuguang Fang, and Robert Caiming Qiu. Performance of an aggregation-based MAC protocol for high-data-rate ultrawideband ad hoc networks. *IEEE Transactions on Vehicular Technology*, 56(1):312–321, 2007.

[54] Samuel Madden, Michael Franklin, and Joseph Hellerstein. TinyDB: An acquisitional query processing system for sensor systems. In *ACM Transactions on Database Systems*, 2004.

[55] Samuel Madden, Michael Franklin, Joseph Hellerstein, and Wei Hong. TAG: a tiny aggregation service for ad-hoc sensor networks. In *OSDI*, 2002.

[56] Victoria Manfredi, Robert Hancock, and Jim Kurose. Robust routing in dynamic manets, 2008.

[57] M. Mohandespour and A.E. Kamal. 1+n protection in polynomial time: A heuristic approach. In *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, pages 1–5, 2010.

[58] Suman Nath, Phillip Gibbons, Srinivasan Seshan, and Zachary Anderson. Synopsis diffusion for robust aggregation in sensor networks. In *ACM SenSys*, 2004.

[59] Yvonne Anne Oswald, Stefan Schmid, and Roger Wattenhofer. Tight bounds for delay-sensitive aggregation. In *ACM PODC*, 2008.

[60] Joon-Sang Park, M. Gerla, D.S. Lun, Y. Yi, and M. Medard. Codecast: a network-coding-based ad hoc multicast protocol. *Wireless Communications, IEEE*, 13(5):76–81, 2006.

[61] Sundeep Pattem, Bhaskar Krishnamachari, and Ramesh Govindan. The impact of spatial correlation on routing with compression in wireless sensor networks. In *ACM/IEEE IPSN*, 2004.

[62] Rishi Saket and Nicolas Navet. Frame packing algorithms for automotive applications. *Journal of Embedded Computing*, 2:93–102, 2006.

[63] Ignacio Solis and Katia Obraczka. The impact of timing in data aggregation for sensor networks. In *IEEE ICC*, 2004.

[64] Anand Srinivas and Eytan Modiano. Minimum energy disjoint path routing in wireless ad-hoc networks. In *Proceedings of the 9th annual international conference on Mobile computing and networking*, MobiCom '03, pages 122–133, New York, NY, USA, 2003. ACM.

[65] J. W. Suurballe. Disjoint paths in a network. *Networks*, 4(2):125–145, 1974.

[66] J. W. Suurballe and R. E. Tarjan. A quick method for finding shortest pairs of disjoint paths. *Networks*, 14(2):325–336, 1984.

[67] Pu Wang, Jun Zheng, and Cheng Li. Data aggregation using distributed lossy source coding in wireless sensor networks. In *IEEE GLOBECOM*, 2007.

[68] Qiao Xiang, Jinhong Xu, Xiaohui Liu, Hongwei Zhang, and Loren J. Rittle. When in-network processing meets time: Complexity and effects of joint optimization in wireless sensor networks. Technical report, Wayne State University (http://www.cs.wayne.edu/~hzhang/group/TR/DNC-TR-09-01.pdf), 2009.

[69] Yong Yao and Johannes Gehrke. The cougar approach to in-network query processing in sensor networks. In *ACM SIGMOD*, 2002.

[70] Zhenzhen Ye, Alhussein A. Abouzeid, and Jing Ai. Optimal policies for distributed data aggregation in wireless sensor networks. In *IEEE INFOCOM*, 2007.

[71] Sunhee Yoon and Cyrus Shahabi. The clustered aggregation (CAG) technique leveraging spatial and temporal correlation in wireless sensor networks. *ACM Transactions on Sensor Networks*, 3(1), 2007.

[72] Yang Yu, Viktor Prasanna, and Bhaskar Krishnamachari. Energy minimization for real-time data gathering in wireless sensor networks. *IEEE Transactions on Wireless Communications*, 5(11), 2006.

[73] Chi Zhang, Yuguang Fang, and Xiaoyan Zhu. Throughput-delay tradeoffs in large-scale manets with network coding. In *INFOCOM 2009, IEEE*, pages 199–207, April 2009.

[74] Xiaolan Zhang, G. Neglia, J. Kurose, and D. Towsley. On the benefits of random linear coding for unicast applications in disruption tolerant networks. In *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, 2006 4th International Symposium on*, pages 1–7, April 2006.

[75] Xiaoyan Zhu, Hao Yue, Fang Yuguang, and Yumin Wang. A batched network coding scheme for wireless networks. *Wireless Networks*, 15(8):1152–1164, Nov 2009.

ABSTRACT

IN-NETWORK PROCESSING FOR MISSION-CRITICAL WIRELESS NETWORKED SENSING AND CONTROL: A REAL-TIME, EFFICIENCY, AND RESILIENCY PERSPECTIVE

by

QIAO XIANG

August 2014

Advisor: Dr. Hongwei Zhang

Major: Computer Science

Degree: Doctor of Philosophy

As wireless cyber-physical systems (WCPS) are increasingly being deployed in mission-critical applications, it becomes imperative that we consider application QoS requirements in in-network processing (INP). In this dissertation, we explore the potentials of two INP methods, packet packing and network coding, on improving network performance while satisfying application QoS requirements. We find that not only can these two techniques increase the energy efficiency, reliability, and throughput of WCPS while satisfying QoS requirements of applications in a relatively static environment, but also they can provide low cost proactive protection against transient node failures in a more dynamic wireless environment.

We first study the problem of jointly optimizing packet packing and the timeliness of data delivery. We identify the conditions under which the problem is strong NP-hard, and we find that the problem complexity heavily depends on aggregation constraints instead of network and traffic properties. For cases when the problem is NP-hard, we show that there is no polynomial-time approximation scheme (PTAS); for cases when the problem can be solved in polynomial time, we design polynomial time, offline algorithms for finding the optimal packet packing schemes. We design a distributed, online protocol *tPack* that schedules packet transmissions to maximize the local utility of packet packing at each node. We evaluate the properties of tPack in NetEye testbed. We find that jointly optimizing data delivery timeliness and packet packing and considering real-world aggregation constraints significantly improve network performance.

We then work on the problem of minimizing the transmission cost of network coding based routing in sensor networks. We propose the first mathematical framework so far as we know on how to theoretically compute the expected transmission cost of NC-based routing in terms of expected number of transmission. Based on this framework, we design a polynomial-time greedy algorithm for forwarder set selection and prove its optimality on transmission cost minimization. We designed EENCR, an energy-efficient NC-based routing protocol that implement our forwarder set selection algorithm to minimize the overall transmission cost. Through comparative study on EENCR and other state-of-the-art routing protocols, we show that EENCR significantly outperforms CTP, MORE and CodeOR in delivery reliability, delivery cost and network goodput.

Furthermore, we study the 1+1 proactive protection problem using network coding. We show that even under a simplified setting, finding two node-disjoint routing braids with minimal total cost is NP-hard. We then design a heuristic algorithm to construct two node-disjoint braids with a transmission cost upper bounded by two shortest node-disjoint paths. And we design ProNCP, a proactive NC-based protection protocol using similar design philosophy as in EENCR. We evaluate the performance of ProNCP under various transient network failure scenarios. Experiment results show that ProNCP is resilient to various network failure scenarios and provides a state performance in terms of reliability, delivery cost and goodput.

Our findings in this dissertation explore the challenges, benefits and solutions in designing real-time, efficient, resilient and QoS-guaranteed wireless cyber-physical systems, and our solutions shed lights for future research on related topics.

AUTOBIOGRAPHICAL STATEMENT

I received the Bachelor Degree of Engineering in Information Security and the Bachelor Degree of Economics in Nankai University, Tianjin, China in 2007. After that, I joined the Department of Computer Science at Wayne State University in Detroit, Michigan as a graduate student. My research interests include protocol design in wireless sensor networks, cyber-physical systems and vehiular networks, smart grid and network economics. I received the Master Degree of Science majored in computer science in 2012. I will be joining the School of Computer Science in McGill University at Montreal in Quebec, Canada as a postdoctoral fellow after finishing my doctoral study.