

# Context-Aware Interflow Network Coding and Scheduling in Wireless Networks

Tuan Tran, *Member, IEEE*, and Thinh Nguyen, *Member, IEEE*

**Abstract**—Recent approaches using network coding (NC) to mix data from different flows show significant throughput improvement in wireless networks. However, in this paper, we argue that exhaustively mixing packets from different flows may decrease network quality of service (QoS), particularly in the presence of flows with different service classes. We therefore propose a *context-aware interflow network coding and scheduling* (CARE) framework, which adaptively encodes data across the traffic to maximize the network QoS. First, we develop a perception-oriented QoS (PQoS) to measure the user satisfaction of different types of services. Next, based on the characteristics of the traffic, we optimally combine data across the flows and schedule the encoded packets in each time frame to maximize the PQoS at the receivers. Solving CARE is NP-hard; thus, we devise a computationally efficient approximation algorithm based on the Markov chain Monte Carlo method to approximate the optimal solution. We prove that the proposed approximation algorithm is guaranteed to converge to the optimal solution. The analytical and simulation results show that, under certain channel conditions, the proposed CARE-based schemes not only improve the network QoS but achieve high throughput across all receivers as well. Additionally, the results show that the approximation algorithm is efficient and robust to the number of data flows. In some transmission conditions, our CARE-based schemes can improve the network QoS up to 50% compared with the existing randomized NC techniques.

**Index Terms**—Multiuser multiservice scheduling, quality of service (QoS), random network coding (RNC), wireless networks.

## I. INTRODUCTION

WE consider the problem of downlink transmission in wireless networks, whereby multiple data flows share a single wireless channel. Traditionally, data transmission is performed via the *store-and-forward* routing protocols in which an intermediate node stores incoming data and forwards them to the neighboring nodes toward the destinations without altering contents of the data. Differently, in the new *network coding* (NC) approach [1], an intermediate node is allowed to

combine the incoming data packets before sending them out to the receivers. It has been shown that NC-based approaches significantly improve network performance, such as throughput [2]–[6], transmission delay [3], [7], and energy [8], [9]. For instance, Nguyen *et al.* [2], [6] showed that XOR-based NC schemes used in conjunction with a scheduler at a WiFi access point (AP) can significantly improve the network throughput of a broadcast session. The authors showed that, under some transmission conditions, bandwidth efficiency could be double compared with the traditional Auto Repeat reQuest (ARQ) approaches [10], [11]. Additionally, Eryilmaz *et al.* [3] have shown that, in unreliable wireless networks, transmission delay decreases substantially by using NC. Furthermore, Tran *et al.* [12], [13] showed that significant bandwidth gain can also be achieved by employing NC in conjunction with channel coding techniques across different unicast sessions. In a different avenue, Wu *et al.* [8] showed that NC can also be used to minimize the transmission energy in mobile ad-hoc networks as well.

In this paper, we focus on using NC-based approaches to improve the quality of service (QoS) of wireless networks that consist of multiple unicast flows. Generally speaking, providing high QoS for multiuser wireless networks is challenging. First, wireless links often suffer due to severe fading and interference. Additionally, channel conditions that usually change over time significantly affect the QoS at the receivers. Furthermore, the heterogeneity of channel conditions makes the scheduling problem more challenging, as receivers usually experience different data losses. Retransmitting lost data to a receiver in a bad channel condition may decrease the network bandwidth efficiency because data could be duplicated at other receivers. On the other hand, only serving receivers in good channel conditions could leave many other receivers in worse channel conditions with unacceptable QoS. The problem usually becomes combinatorially hard in nature.

In fact, there exists literature on using NC to improve network QoS. Notably, Seferoglu *et al.* [14], [15] proposed video-aware opportunistic XOR-based network encoding and scheduling schemes for video streaming. The proposed schemes take into account both video distortion and deadlines of the packets for optimal data encoding and scheduling. The simulation results showed significant video quality improvement [i.e., peak-signal-to-noise ratio (PSNR)] compared with the approaches without video-aware encoding. Those works, however, considered only the case where all data flows are multimedia streams (i.e., video). As a result, they may not work well when applied directly to scenarios where traffic includes both delay-sensitive (e.g., video streaming) and elastic applications (e.g., web browsing). In such a setting, the performance metric of

Manuscript received October 31, 2014; revised April 7, 2015, July 20, 2015, and November 17, 2015; accepted January 1, 2016. This work was supported in part by the National Science Foundation under Grant CNS-0845476 and Grant CNS-1547450 and in part by the Faculty Scholarship Grant of Sullivan University. This paper was presented in part at the 19th IEEE International Conference on Computer Communications and Networks (ICCCN), 2010. The review of this paper was coordinated by Prof. C. Assi.

T. Tran is with the College of Information and Computer Technology, Sullivan University, Louisville, KY 40205 USA (e-mail: ttran@sullivan.edu).

T. Nguyen is with the School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR 97331 USA (e-mail: thinhq@eecs.oregonstate.edu).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TVT.2016.2520361

88 delay-sensitive applications (e.g., PSNR for video streaming)  
 89 might not be a good metric to measure the QoS of the delay-  
 90 tolerant applications. Generally, the heterogeneity of the traffic  
 91 content makes the encoding and scheduling problem more  
 92 challenging because the transmitter needs to consider not only  
 93 the transmission deadline but the *characteristics of the traffic*  
 94 as well.

95 We consider the problem of multiuser downlink transmission  
 96 of heterogeneous traffic in lossy wireless networks. A typical  
 97 approach for such a system is to overprovision the QoS require-  
 98 ments for various flows. Such a system will work well as long  
 99 as the aggregate demand from all the flows does not exceed the  
 100 network capacity [16]. However, in an overloaded transmission  
 101 scenario, the system performance is likely to deteriorate rapidly  
 102 due to a traffic surge [17]. Therefore, we propose a new  
 103 NC-based framework called Context-aware Interflow Net-  
 104 work Coding and Scheduling (CARE) which utilizes the user  
 105 perception-oriented QoS model (PQoS) [18] for optimal inter-  
 106 flow data encoding and scheduling. Generally speaking, CARE  
 107 optimally encodes and schedules data transmission based on  
 108 both channel conditions and characteristics of the traffic to  
 109 maximize the network QoS from the user's perspective. Our  
 110 results show that CARE significantly improves the network  
 111 QoS and throughput, in comparison with the state-of-the-art  
 112 NC-based approaches. The performance improvement basically  
 113 comes from the optimal trade-off between the number of useful  
 114 data packets transmitted by the deadline and the bandwidth  
 115 allocation to different flows via its PQoS objective function.  
 116 To the best of our knowledge, this is one of a few papers  
 117 that consider PQoS as the objective function in formulating the  
 118 multiuser downlink scheduling using NC in wireless networks.  
 119 Extending from our preliminary results in [19], the main  
 120 contributions of this paper are summarized as follows.

- 121 • We first show that the approaches optimized for through-  
 122 put might decrease the QoS at some receivers in the  
 123 presence of traffic with different service classes. We then  
 124 devise PQoS as the network metric to quantitatively mea-  
 125 sure the performance of different transmission strategies.  
 126 The CARE framework based on PQoS consolidates not  
 127 only the traffic priorities but also the characteristics of the  
 128 flows in its encoding and scheduling operation. Analyses  
 129 on QoS performance for different transmission strategies  
 130 are provided in detail.
- 131 • We formulate CARE as a combinatoric optimization  
 132 problem with constraints. We further show that CARE  
 133 can be reduced to a set of weighted stochastic knapsack  
 134 problems and, therefore, is NP-hard [20]. We then exploit  
 135 the traffic characteristics to devise an efficient approxima-  
 136 tion algorithm based on the Markov chain Monte Carlo  
 137 (MCMC) method for finding a near-optimal solution. Our  
 138 results show that, under certain channel conditions and  
 139 QoS requirements, our proposed CARE-based schemes  
 140 lead to significant network performance improvement  
 141 for both delay-sensitive and delay-tolerant data flows.  
 142 We further provide analytical results on the asymptotic  
 143 behavior and upper bound on the convergence time of  
 144 the approximation algorithm based on the canonical path

technique [21]. We also describe context-aware partial  
 interflow NC (PCARE), an extension of CARE, which  
 allows for finer grained bandwidth allocation.

- We provided theoretical analysis and intensive simula-  
 tions to elaborate the system performance improvement of  
 our proposed schemes. Our results reveal that optimizing  
 the PQoS is equivalent to optimizing the network effective  
 throughput constrained on the fairness condition among  
 the users. The results also show that the approximation  
 algorithm is efficient and robust to the number of data  
 flows and quickly converges to the optimal solution.

The remainder of this paper is organized as follows. We first  
 discuss some background and related work in Section II. In  
 Section III, we describe the system model, the issues of the  
 existing approaches, and performance metric. In Section IV,  
 we analyze the system performance of different transmission  
 strategies, CARE formulation, and its hardness. In Section V,  
 we develop an approximation algorithm for CARE. Simula-  
 tions and discussions are provided in Section VI. Finally, we  
 conclude this paper in Section VII.

## II. BACKGROUND AND RELATED WORK

*Random Network Coding:* The notion of NC, i.e., mixing  
 of data at intermediate nodes to increase the overall multicast  
 throughput of a network, was first proposed in the seminal paper  
 by Ahlswede *et al.* [1]. Its key idea is to prove the existence of  
 some network codes (method of mixing data at intermediate  
 nodes) that achieve multicast capacity. This spurred a number  
 of works on construction of practical network codes, including  
 algebraic, algorithmic, and randomized approaches [22]–[25].  
 In particular, the work in [22] proposed a class of linear network  
 codes for multicast transmission. The author proved that linear  
 coding suffices to achieve the maximum throughput, which is  
 the max-flow min-cut from the source to each receiving node.  
 Inspired by this work, Koetter and Medard [23] proposed a  
 theoretical framework based on algebraic tools for deriving  
 the conditions to achieve capacity in networks using linear  
 codes. The proposed framework shows interesting connections  
 between certain systems of polynomial equations and the so-  
 lutions to network routing problems. Notably, the work in [4]  
 (and its extended version in [26]) proposed a random NC  
 (RNC) framework for efficient network code construction in  
 a distributed manner. In particular, it shows that intermediate  
 nodes in a network do not need to cooperate with each other to  
 generate coded packets. Instead, each node independently gen-  
 erates its coded packets by combining its incoming data with  
 the coding coefficients randomly selected from a large finite  
 field. The authors proved that, with probability approaching 1,  
 the encoded packets are independent. Based on this result,  
 Chou *et al.* [25] proposed a practical solution to implement  
 NC for an arbitrary network topology. In particular, it has been  
 shown that, by inserting the coding coefficients into the coded  
 packets' headers, the sinks can reconstruct the original data  
 efficiently by solving a system of linear equations constructed  
 by the encoded data.

Time slot	1		2		3		4
Scheme	UNI	RNC	UNI	RNC	UNI	RNC	UNI
Packet	$a$	$c_1$	$b$	$c_2$	$a$	$c_3$	$b$
$D_1$	x	x	-	o	o	o	-
$D_2$	-	o	x	x	-	o	o

Fig. 1. Example of UNI and RNC transmission schemes for two receiver scenarios. For the RNC scheme, coded packets  $c_i = \alpha_i a + \beta_i b$ . Packet receipts are denoted by “x,” “o,” and “-” for lost, successful, and “received but useless” packets, respectively.

199 *RNC for Wireless Networks*: It has been shown that applying  
 200 NC in wireless ad-hoc networks can significantly improve the  
 201 network bandwidth efficiency [27], transmission delay [28],  
 202 [29], or transmission energy [7]. The key idea of these works  
 203 is to exploit the nature of a broadcast signal, which can be  
 204 intercepted by many neighboring nodes. Data of different flows  
 205 are then combined (i.e., performing RNC) together to generate  
 206 coded data packets before sending them to other nodes in  
 207 the vicinity. Recently, Douik *et al.* [30], [31] have proposed  
 208 techniques to reduce the decoding delay for different feedback  
 209 constraints. These works, however, only focus on broadcasting  
 210 transmission and do not consider the flows with heterogeneous  
 211 content and services. A detailed list of wireless applications  
 212 beneficial from using NC in these settings can be found in [29].  
 213 Our network model is closest to the model used in [2] and  
 214 [6], whereby the authors model a broadcast session in a last-  
 215 mile network. Differently, we consider multiple unicast flows  
 216 sharing a bandwidth-constrained channel. Our transmission  
 217 model can be applied to many practical transmission scenarios  
 218 in WLAN, WiMAX, and cellular networks. Before describing  
 219 our system model, we first illustrate the benefit of using NC in  
 220 such a setting.

221 *Example 1*: An AP wishes to send two packets  $a$  and  $b$  to  
 222 two receivers  $D_1$  and  $D_2$ , respectively. In ARQ unicast protocol  
 223 (UNI), the AP uses the first and second time slots to send  $a$   
 224 and  $b$ , respectively. As illustrated in Fig. 1, packet  $a$  is lost at  
 225  $D_1$  while successfully received at  $D_2$ . However,  $D_2$  discards  $a$   
 226 because it wants packet  $b$  instead. Similarly, in the second time  
 227 slot, packet  $b$  is successfully received by  $D_1$  (but it is discarded  
 228 because  $D_1$  wants  $a$  instead) while lost at  $D_2$ . Assuming that,  
 229 in the next two time slots, the transmission links are in good  
 230 conditions, then the AP can successfully retransmit the lost  
 231 data packets to their intended receivers. As a result, it needs  
 232 four time slots to deliver two packets  $a$  and  $b$  to  $D_1$  and  $D_2$ ,  
 233 respectively.

234 Next, we consider a transmission scheme using RNC, as  
 235 proposed in [32]. In this approach, the AP generates coded  
 236 packets by linearly combining  $a$  and  $b$  with random coefficients  
 237 and sending them out to the receivers. For example, coded  
 238 packets  $c_i$  are generated as  $c_i = \alpha_i a + \beta_i b$ ,  $i = \{1, 2, 3, \dots\}$ ,  
 239 where  $\alpha_i$  and  $\beta_i$  are coefficients drawn randomly from a large  
 240 finite field  $F_q$  ( $q$  is the field size). The AP then broadcasts  
 241 two coded packets  $c_1$  and  $c_2$  in the first and second time  
 242 slots, respectively, as shown in Fig. 1. As a result,  $D_1$  and  
 243  $D_2$  will successfully receive  $c_2$  and  $c_1$ , respectively. In the  
 244 third time slot, the AP broadcasts another coded packet  $c_3$ ,  
 245 which is successfully received by both receivers. After three

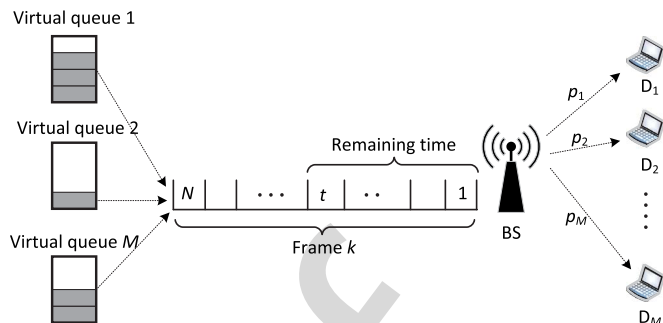


Fig. 2. System model. At the beginning of frame  $k$ , new arrived packets will be scheduled for transmission in the next  $N$  time slots (period of a frame).

246 transmissions, each of the receivers now has two coded packets,  
 247 which can be used to recover their desired data by solving  
 248 a system of linear equations using the encoded packets. We  
 249 note that the coefficients are included in the packets' headers,  
 250 enabling the receivers to solve the linear equation system.  
 251 It requires additional transmission overhead; however, with a  
 252 sufficiently large packet size, this overhead is negligible [25].  
 253 Thus, the RNC approach needs only three transmissions to  
 254 deliver two packets to the receivers, saving 25% transmission  
 255 bandwidth compared with the UNI scheme.

### III. SYSTEM MODEL, ISSUES, AND PERFORMANCE METRIC

#### A. System Model

256 We consider the problem of time-slotted downlink transmis-  
 257 sion of  $M$  data flows to  $M$  receivers (users) in lossy wireless  
 258 networks, as illustrated in Fig. 2. We assume that the transmitter  
 259 uses  $M$  “virtual queues” to store randomly arrived packets of  
 260 different flows before sending them out to the corresponding  
 261 receivers. Transmissions are scheduled by frames (periods),  
 262 each consisting of  $N$  time slots. Generally, the value of  $N$   
 263 can be determined by the hard deadline of buffered data or  
 264 by the number of backlogged data packets in the buffer [15],  
 265 [33]–[35]. When a new flow arrives at the transmitter in the  
 266 current transmission frame, it will be scheduled for transmis-  
 267 sion in the next frame. Without loss of generality, we assume  
 268 that there are  $k_i$ ,  $i = 1, 2, \dots, M$  data packets being delivered  
 269 to receiver  $D_i$  in each transmission frame. In this paper, we  
 270 focus on finding an optimal flow partition scheme for encoding  
 271 and scheduling data across different flows, given a set of data  
 272 packets for each transmission frame.

273 We assume that the scheduled packets of delay-sensitive  
 274 applications, which cannot be delivered to the corresponding  
 275 receivers by the deadline, will be discarded from the system.  
 276 The system QoS is computed based only on the number of data  
 277 packets that have been received successfully within that period.  
 278 On the other hand, for delay-tolerant reliable applications such  
 279 as file transfer, the lost packets will be retransmitted until they  
 280 are successfully delivered (possibly via multiple transmission  
 281 frames). Detail of the mathematical formula used to compute  
 282 the system QoS is defined in Section III-C3. In addition, we  
 283 assume that the transmission links between the transmitter and  
 284

287 the receivers are heterogeneous and independent identically  
 288 distributed binary erasure channels with erasure probability  $p_i$ .  
 289 A flow  $i$  implements a packet-level forward error-correcting  
 290 (FEC) code  $(n_i, k_i)$  (e.g., [36]) to cope with data errors. Using  
 291 such an FEC code, receiving  $k_i$  out of  $n_i$  transmitted packets is  
 292 sufficient for receiver  $D_i$  to recover the original information.  
 293 The code rates  $k_i/n_i$  are prespecified based on the network  
 294 conditions or/and priority of the users' subscription. Finding  
 295 the optimal coding rates is beyond the scope of this paper. Ad-  
 296 ditionally, we assume that the transmitter has enough memory  
 297 to store data for at least one transmission frame  $N = \sum_{i=1}^M n_i$ .  
 298 Furthermore, in our model, each receiver maintains a *decod-*  
 299 *ing buffer* for storing coded packets received within a frame.  
 300 By the end of each data frame, a receiver will decode the  
 301 received packets and push them to the upper Open Systems  
 302 Interconnection (OSI) layer above for further processing. It is  
 303 important to note that the use of a decoding buffer at receivers  
 304 is a standard assumption, which has been adopted for several  
 305 years in NC-based techniques and literature (e.g., [5], [6], [15],  
 306 [26], [30], [37], [38], and references therein).

### 307 B. What Could Go Wrong With Interflow Network Coding?

308 As observed in *Example 1*, mixing packets from different  
 309 flows is clearly beneficial. However, *should one advocate*  
 310 *mixing at every opportunity?* The answer should be "No." In  
 311 particular, Wu *et al.* were the first to consider this mixing  
 312 problem in a different context [39]. Intuitively, exhaustively  
 313 mixing the data of several flows may decrease the chance that a  
 314 receiver can recover its information. To illustrate this point, we  
 315 show a simple counterexample as follows.

316 *Counterexample:* We consider the same setup, as shown in  
 317 Fig. 1. In addition, we assume that packet losses at  $D_1$  and  $D_2$   
 318 are independent and follow the Bernoulli trial with  $p_1 = 1/3$   
 319 and  $p_2 = 2/3$ , respectively. To transmit data reliably, it makes  
 320 sense for the transmitter to employ stronger protection for the  
 321 data intended to  $D_2$ . Thus, suppose that two packet-level FEC  
 322 codes  $(n_1, k_1) = (3, 2)$  and  $(n_2, k_2) = (3, 1)$  are used for the  
 323 flows to  $D_1$  and  $D_2$ , respectively.<sup>1</sup> We recall that, by using  
 324 a code  $(n, k)$ , the transmitter uses  $n$  time slots to transmit  $k$   
 325 information packets ( $n \geq k$ ), whereby receiving any  $k$  packets  
 326 out of the  $n$  transmitted packets is sufficient to recover the  $k$   
 327 original packets. Suppose that the transmitter has  $n_1 + n_2 = 6$   
 328 time slots for delivering three packets, i.e., two for  $D_1$  and one  
 329 for  $D_2$ . In a non-mixing technique, i.e., packets from different  
 330 flows are sent separately, the probability that all the receivers  
 331 recover their desired data is computed as

$$P_{\text{UNI}} = \prod_{i=1}^2 \sum_{j=0}^{r_i} \binom{n_i}{j} p_i^j (1-p_i)^{n_i-j} \quad (1)$$

332 where  $r_i = n_i - k_i$  for  $i = \{1, 2\}$ . Substituting values of  $p_i$ ,  $n_i$ ,  
 333 and  $k_i$  into (1), we have  $P_{\text{UNI}} = 0.5213$ , which is about 1/2  
 334 packet per time slot.

On the other hand, in an RNC-based technique, all packets 335  
 are mixed to produce coded packets. In such a transmission 336  
 strategy, each receiver needs to receive at least three coded 337  
 packets correctly, to recover its desired information [41]. The 338  
 transmitter has six time slots for delivering the *coded* packets 339  
 to both receivers. The probability that both receivers recover 340  
 their desired data is given by 341

$$P_{\text{RNC}} = \prod_{i=1}^2 \sum_{j=0}^r \binom{N}{j} p_i^j (1-p_i)^{N-j} \quad (2)$$

where  $r = \sum_{i=1}^2 (n_i - k_i) = 3$  and  $N = \sum_{i=1}^2 n_i = 6$ . Sub- 342  
 stituting the values of  $p_i$  for  $i = \{1, 2\}$  into (2), we obtain 343  
 $P_{\text{RNC}} = 0.2876$ . This is approximately equivalent to a through- 344  
 put of 1/5 packet per time slot, which is about 2.5 times less than 345  
 that of the non-NC technique earlier. Obviously, applying NC 346  
 in this case decreases the network throughput. 347

### C. Perception-Oriented QoS 348

We next describe how the PQoS metric is constructed to 349  
 measure the performance of different transmission strategies. 350  
 Roughly speaking, PQoS function is used to estimate the ser- 351  
 vice satisfaction at each user. Thus, PQoS depends not only on 352  
 the number of received packets within a time period but also 353  
 on the type of service (ToS). For the sake of exposition, we 354  
 categorize the network traffic into two types: delay-sensitive 355  
 traffic (e.g., video streaming, audio streaming, etc.) and elastic 356  
 traffic (e.g., file transfer, e-mail, etc.).<sup>2</sup> We note, however, that 357  
 one can easily extend the framework to the cases of more than 358  
 two types of traffic, albeit a more sophisticated model. 359

1) *PQoS for Delay-Sensitive Traffic:* In our model, delay- 360  
 sensitive traffic (e.g., video streaming) is encoded into multiple 361  
 layers, e.g., a base layer and enhancement layers [42], [43]. 362  
 In such a model, the base layer must be presented to present 363  
 other enhancement layers. Thus, to maintain a minimal QoS, a 364  
 receiver needs to receive at least  $N_0$  packets of the base layer 365  
 per transmission frame. When the number of received packets is 366  
 fewer than  $N_0$ , the QoS at the receiver decreases significantly. 367  
 The reason is that, in such an encoding scheme, the QoS at 368  
 the receiver is mainly contributed by the base layer. On the 369  
 other hand, when more packets of the enhancement layers 370  
 are received, the QoS at the receiver only increases slightly 371  
 due to only additional details of the information added into 372  
 the base frame. We extend the satisfaction function proposed 373  
 in [18] to model the PQoS for delay-sensitive applications. 374  
 Mathematically, we can represent it in (3), shown at the bottom 375  
 of the next page, where  $S_i$  denotes the number of packets 376  
 received successfully, and  $N_0$  denotes the minimum number of 377  
 packets to maintain a satisfaction factor of  $\gamma_0 \in [0, 1]$ . In this 378  
 formulation, the first case indicates that, when the number of 379  
 received packets is fewer than the threshold  $\gamma_0 N_0$ , the value 380  
 of PQoS is equal to zero. The intuition can be reasoned in the 381  
 context of layered video transmission (e.g., MPEG-2) where the 382  
 base layer is unrecoverable. Thus, no information is displayed, 383

<sup>1</sup>We note that the packet-level FEC codes here can be viewed as the raptor codes [40] with  $n_i$  and  $k_i$ , respectively, being the output and original symbols.

<sup>2</sup>We use the terms "elastic" and "delay-tolerant" interchangeably.

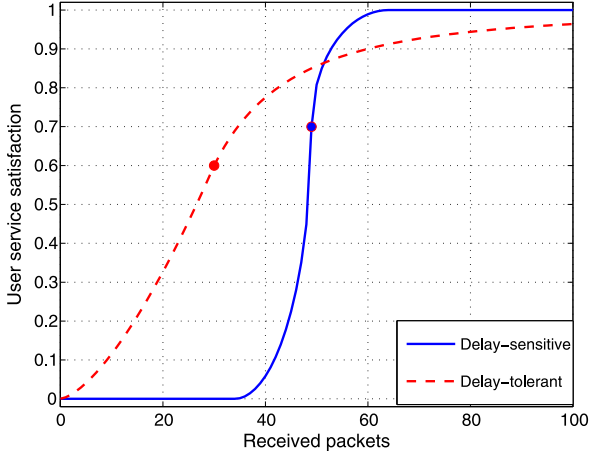


Fig. 3. Satisfaction functions of different applications versus the number of received packets.

384 resulting in a zero PQoS. The second case accounts for the  
 385 scenario when the number of received packets is greater than  
 386 the minimum threshold  $\gamma_0 N_0$  but less than  $N_0$ . In this case,  
 387 the PQoS steeply decreases due to only a part of the base  
 388 layer recovered. The third case indicates that the value of PQoS  
 389 increases significantly when the number of received packets is  
 390 greater than  $N_0$ . In this case, more detail of the video frame is  
 391 added to achieve high-definition display. Finally, the last case  
 392 accounts for a scenario wherein the missing data add negligible  
 393 QoS to the data flow, resulting in a high value of PQoS close  
 394 to one. An example of the PQoS function for delay-sensitive  
 395 application, with respect to the number of received packets, is  
 396 illustrated by the blue solid curve in Fig. 3.

397 2) *PQoS for Delay-Tolerant Traffic*: We use different func-  
 398 tions to model elastic traffic, as illustrated by the red dashed  
 399 curve in Fig. 3. For such a flow, the parameter  $N_0$ , i.e., the  
 400 minimum data received per frame, is viewed as the minimum  
 401 bandwidth allocated to that flow. When the number of received  
 402 packets is fewer or greater than  $N_0$ , respectively, the satis-  
 403 faction function decreases or increases slightly. We note that,  
 404 in delay-tolerant traffic, if a transmitted packet is lost during  
 405 transmission, a negative acknowledgement (NACK) will be sent  
 406 from the receiver to the transmitter for retransmission (possibly  
 407 in other frames). Thus, for reliable data flows (e.g., data file  
 408 downloading), every byte of the information will be reliably  
 409 delivered to the receiver. Mathematically, the PQoS for elastic  
 410 traffic is given as

$$\gamma_i = \mathcal{F}(S_i) = \begin{cases} \gamma_0 \cdot \left(\frac{S_i}{N_0}\right)^2, & S_i < N_0 \\ 1 - (1 - \gamma_0) \left(\frac{N_0}{S_i}\right)^2, & S_i \geq N_0. \end{cases} \quad (4)$$

In the first case, when the number of received packets is fewer  
 411 than  $N_0$ , the value of PQoS decreases slightly by a factor of  
 412 the ratio of  $S_i$  and  $N_0$ . It is worth noting that this ratio is less  
 413 than one; thus, its square would result in a smaller value. On  
 414 the other hand, when  $S_i \geq N_0$ , PQoS increases slightly. The  
 415 intuition of the proposed PQoS functions is to reflect the fact  
 416 that a slightly longer or shorter delay in receiving a delay-  
 417 tolerant data packet, e.g., e-mail, does not affect much to the  
 418 user's perception of the QoS. 419

3) *Performance Metric*: We use the average network PQoS  
 420 as our metric to compare the performance of different trans-  
 421 mission strategies. The average network PQoS is computed by  
 422 averaging the expected PQoS across all the users for each trans-  
 423 mission frame over infinite system runtime. Mathematically, we  
 424 have that 425

$$\gamma = \lim_{\tau \rightarrow \infty} \frac{1}{\tau M} \sum_{\tau} \sum_{i=1}^M c_i \mathbb{E}[\gamma_i] \quad (5)$$

where  $\tau$  is the operation time of the network divided into  
 426 several transmission frames,  $M$  is the number of data flows,  
 427  $c_i$  is the flow priority, and  $\mathbb{E}[\gamma_i]$  denotes the expected PQoS  
 428 user  $i$  in one transmission frame. The  $\lim_{\tau \rightarrow \infty}$  indicates that  
 429 the expectation of the system performance is computed over a  
 430 long time interval. Given the satisfaction functions, we call a  
 431 transmission technique the best, if it has the largest expected  
 432 PQoS across all users. 433

#### IV. PERFORMANCE ANALYSIS 434

In this paper, we analyze different transmission strategies,  
 435 depending on how the transmitter exploits the characteristics  
 436 of the traffic. In particular, we analyze the performance of the  
 437 traditional UNI, systematic RNC (SRNC), and CARE. 438

##### A. Unicast 439

In this technique, data packets of different information flows  
 440 are transmitted separately in a round-robin fashion [44]. In  
 441 each period (frame), the transmitter allocates  $n_i$  time slots to  
 442 transmit  $k_i$  data packets to receiver  $D_i$  [i.e., packet-level FEC  
 443 code  $(n_i, k_i)$ ]. If there is a packet loss, the transmitter uses the  
 444  $n_i - k_i$  redundant time slots to retransmit the lost packets. The  
 445 transmitter switches to transmit data packets for other users,  
 446 when it receives an acknowledgment (ACK) message from  $D_i$   
 447 indicating that all data have been received successfully, or all  
 448 time slots allocated for it have been used. Considering flow  $i$   
 449 destined to receiver  $D_i$  and letting  $p_i$  denote the packet erasure  
 450

$$\gamma_i = \mathcal{F}(S_i) = \begin{cases} 0, & S_i < \gamma_0 N_0 \\ \gamma_0 \left(1 - \frac{\sqrt{(N_0 - S_i)(N_0 + S_i - 2\gamma_0 N_0)}}{(1 - \gamma_0) N_0}\right), & \gamma_0 N_0 \leq S_i < N_0 \\ \gamma_0 + (1 - \gamma_0) \frac{\sqrt{(S_i - N_0)[N_0 - S_i + 2N_0(1 - \gamma_0)]}}{(1 - \gamma_0) N_0}, & N_0 \leq S_i < (2 - \gamma_0) N_0 \\ 1, & S_i \geq (2 - \gamma_0) N_0 \end{cases} \quad (3)$$

451 probability of the transmission link, the probability that receiver  
452  $D_i$  recovers all its data can be written as

$$P_i^s = \sum_{j=k_i}^{n_i} \binom{n_i}{j} p_i^{n_i-j} (1-p_i)^j \quad (6)$$

453 where  $\binom{n_i}{j}$  denotes the number of combinations of size  $j$  out  
454 of  $n_i$  elements. We further note that, by using UNI, i.e., data  
455 transmitted separately, receiver  $D_i$  could recover a fraction of  
456 the original data, if the number of received packets is fewer  
457 than  $k_i$ . Let the random variables  $X$  and  $Y$ , respectively, be  
458 the number of original packets and the total number of received  
459 packets. The probability that  $D_i$  obtains only  $m$  original packets  
460 and the total received packets is fewer than  $k_i$  is given as

$$\begin{aligned} P_i(m) &= P(X = m, Y < k_i) \\ &\stackrel{(a)}{=} \sum_{l=m}^{k_i-1} \Pr(Y = l | X = m) \Pr(X = m). \end{aligned} \quad (7)$$

461 In this case,  $Y$  is equal to  $X$  plus the number of coded  
462 packets received during the second stage transmission while  
463 the sum runs over all the possible values of  $Y$ . We note that  
464 step (a) consists of two parts: 1) the conditional probability  
465 of receiving  $l = m, \dots, k_i - 1$  data packets in total, given  $m$   
466 original data packets received; 2) the probability that  $m$  of  $k_i$   
467 original data packets are received. In the extreme case, i.e.,  
468  $l = m$ , it implies that none of the coded packets is received.  
469 On the other hand, the maximum number of coded packets is  
470  $k_i - 1 - m$ , corresponding to the case of  $k_i - 1$  data packets  
471 received in total. Furthermore, the probability that only  $m$  of  $k_i$   
472 original data packets are received is written as

$$\Pr(X = m) = \binom{k_i}{m} (1-p_i)^m p_i^{k_i-m}. \quad (8)$$

473 Given that only  $m$  original data packets are received, the  
474 probability that only  $l < k_i$  data packets are received over  $n_i$   
475 transmissions is given by

$$\Pr(Y = l | X = m) = \binom{n_i - k_i}{l - m} (1-p_i)^{l-m} p_i^{n_i - k_i - (l-m)}. \quad (9)$$

476 Combining (8) and (9), we have that

$$\begin{aligned} &P(Y = l | X = m) P(X = m) \\ &= \binom{k_i}{m} (1-p_i)^m p_i^{k_i-m} \binom{n_i - k_i}{l - m} \\ &\quad \times (1-p_i)^{l-m} p_i^{n_i - k_i - (l-m)} \\ &= \binom{k_i}{m} \binom{n_i - k_i}{l - m} (1-p_i)^l p_i^{n_i - l}. \end{aligned} \quad (10)$$

477 Therefore, the probability that the  $m$  original data packets and  
478 the total  $l < k_i$  data packets are received can be computed as

$$P_i(m) = \sum_{l=m}^{k_i-1} \binom{n_i - k_i}{l - m} \binom{k_i}{m} p_i^{n_i - l} (1-p_i)^l. \quad (11)$$

Let  $\gamma_i$  denote the PQoS value of receiver  $D_i$ ; therefore, from  
(6) and (7), the expected PQoS across all the users can be  
written as

$$\begin{aligned} \gamma &= \frac{1}{M} \mathbb{E} \left[ \sum_{i=1}^M c_i \gamma_i \right] = \frac{1}{M} \sum_{i=1}^M c_i \mathbb{E}[\gamma_i] \\ &= \frac{1}{M} \sum_{i=1}^M c_i \left( \mathcal{F}(k_i) P_i^s + \sum_{m=0}^{k_i-1} \mathcal{F}(m) P_i(m) \right) \end{aligned} \quad (12)$$

where  $c_i \in (0, 1]$  denotes the weighted factor (priority) for the  
 $i$ th flow (this factor can be justified via the cost that the user  
 $D_i$  pays to the service provider);  $\mathbb{E}[\cdot]$  denotes the expected  
function; and  $\mathcal{F}(\cdot)$  is defined in (3) and (4), depending on the  
type of the data flow.

## B. Systematic Random Network Coding

Transmission in SRNC is classified into base and augmen-  
tation phases. In the base phase, all  $\sum_{i=1}^M k_i$  original packets  
will be transmitted. The receivers cache all the received packets,  
including packets that are not intended to them. By keeping data  
packets intended to others, a receiver can use them to decode  
its desired data in the second phase. Next, in the augmentation  
phase, the transmitter combines the data packets of all flows  
to generate coded packets and broadcasts them to the receivers  
over  $N - \sum_{i=1}^M k_i$  redundant time slots. The intuition of using  
systematic coding that transmits data in two phases is that  
the receivers that lose some original packets can receive more  
coded packets to decode their own data using the RNC method  
[41]. On the contrary, receivers that are unable to obtain a full  
set of data packets can still recover partial data from the original  
packets transmitted in the base phase. Such a transmission has  
been discussed in [45], and generally, it will result in higher  
performance compared to the RNC.

In SRNC, a receiver  $D_i$  can recover  $k_i$  desired packets, if  
it correctly receives either all  $k_i$  original packets or a full set  
of  $K = \sum_{i=1}^M k_i$  packets (either original or coded packets). Let  
 $P_i^s$  denote the probability that  $D_i$  can recover all its data, and  
let the random variables  $U$  and  $V$  denote the original and total  
received packets at receiver  $D_i$ , respectively. We have that

$$\begin{aligned} P_i^s &= P(U = k_i, V < K) + P(U \leq k_i, V \geq K) \\ &= (1-p_i)^{k_i} \left[ \sum_{l=0}^{K-k_i-1} \binom{N-k_i}{l} p_i^{N-k_i-l} (1-p_i)^l \right] \\ &\quad + \sum_{j=0}^{k_i} \binom{k_i}{j} p_i^{k_i-j} (1-p_i)^j \\ &\quad \times \sum_{t=K-j}^{N-k_i} \binom{N-k_i}{t} p_i^{N-k_i-t} (1-p_i)^t. \end{aligned} \quad (13)$$

In (13), the first term accounts for the case when all the  
original data packets are received successfully during the base  
phase transmission. The second term expresses the probability  
that  $j$  original packets are received during the base phase and



515  $t$  coded packets are received during the augmentation phase,  
 516 where  $t = K - j, \dots, N - k_i$ . In this case, at least  $K$  data  
 517 packets (both original and coded packets) are received by  $D_i$ ;  
 518 thus, it can recover its data by solving the system of linear  
 519 equations formed by the received data packets.

520 We further note that  $D_i$  may receive only partial of the  
 521 original data during the base phase. We denote  $P_i(m)$  being  
 522 the probability that  $D_i$  receives  $m$  original packets ( $m < k_i$ ),  
 523 and its total number of packets received correctly is less than  
 524  $K$ . Similar to (7), the probability of partial data recovery of  $D_i$   
 525 is written by

$$\begin{aligned} P_i(m) &= P(U = m, V < K) \\ &= \sum_{l=m}^{K-1} \binom{N-k_i}{l-m} \binom{k_i}{m} p_i^{N-l} (1-p_i)^l. \end{aligned} \quad (14)$$

526 In such cases,  $D_i$  cannot recover all its data, and only data  
 527 received during the base phase contribute to the QoS of flow  
 528  $i$ . From (13) and (14), we express the expected PQoS across all  
 529 receivers as follows:

$$\begin{aligned} \gamma &= \frac{1}{M} \mathbb{E} \left[ \sum_{i=1}^M c_i \gamma_i \right] = \frac{1}{M} \sum_{i=1}^M c_i \mathbb{E}[\gamma_i] \\ &= \frac{1}{M} \sum_{i=1}^M c_i \left( \mathcal{F}(k_i) P_i^s + \sum_{m=0}^{k_i-1} \mathcal{F}(m) P_i(m) \right). \end{aligned} \quad (15)$$

### 530 C. Proposed Context-Aware Interflow 531 Network Coding and Scheduling

532 1) *Main Idea*: Instead of combining all flows together, the  
 533 transmitter now selectively chooses the flows to be mixed based  
 534 on the channel conditions, ToS, and priorities of the flows. We  
 535 assume that the  $M$  incoming data flows are partitioned into  
 536  $G$  groups; then, for each group, the transmitter uses SRNC to  
 537 transmit data to the receivers within that group. The objective of  
 538 the transmitter is to determine the optimal partition (i.e., which  
 539 flows are combined together) to maximize the PQoS across all  
 540 users. It is clear that mixing packets from all incoming flows  
 541 could decrease the system performance due to *mismatch in ToS*,  
 542 *priorities, and channel conditions*. On the other hand, mixing  
 543 packets of flows with similar characteristics could increase the  
 544 network performance. A precise mathematical formulation of  
 545 CARE will be described as follows.

546 2) *CARE Formulation*: Let  $\mathcal{G}$  denote a partition of the  
 547 incoming information flows and  $|\mathcal{G}|$  denote the number of  
 548 groups in  $\mathcal{G}$ . Let  $M_i$  denote the number of data flows in group  
 549  $i$ ,  $i = 1, \dots, |\mathcal{G}|$ . Per each group, we use the SRNC technique  
 550 described earlier to transmit the data. Consider the  $i$ th group,  
 551 and let  $N_i = \sum_{j=0}^{M_i} n_{ij}$  and  $K_i = \sum_{j=0}^{M_i} k_{ij}$ , respectively, de-  
 552 note the total number of available time slots and information  
 553 packets being transmitted for group  $i$ . Here,  $(n_{ij}, k_{ij})$  denotes  
 554 the packet-level FEC of flow delivered to receiver  $j$  of group  $i$ .  
 555 We note that flow  $j$  of group  $i$ , i.e.,  $f_{ij}$ , is nothing but just some  
 556 original data flow  $f_t$ , where the index has been relabeled. Thus,  
 557 the FEC code  $(n_{ij}, k_{ij})$  is also just a relabeled version of the  
 558 original FEC code  $(n_t, k_t)$ . Similarly, as computed in SRNC

technique, the probability that receiver  $j$  of group  $i$ , i.e.,  $D_{ij}$ ,  
 can recover its desired data is given as

$$\begin{aligned} P_{ij}^s &= (1-p_{ij})^{k_{ij}} \left[ \sum_{l=0}^{K_i-k_{ij}-1} \binom{N_i-k_{ij}}{l} p_{ij}^{N_i-k_{ij}-l} (1-p_{ij})^l \right] \\ &+ \sum_{s=0}^{k_{ij}} \binom{k_{ij}}{s} p_{ij}^{k_{ij}-s} (1-p_{ij})^s \\ &\times \sum_{t=K_i-s}^{N_i-k_{ij}} \binom{N_i-k_{ij}}{t} p_{ij}^{N_i-k_{ij}-t} (1-p_{ij})^t. \end{aligned} \quad (16)$$

In this equation, the first term accounts for the case where  
 original packets are successfully received during the basis trans-  
 mission phase of group  $i$ , whereas the second term expresses the  
 probability that at least  $K_i$  data packets (including both original  
 and coded packets) are received successfully after both phases  
 of transmission.

Similarly, it is straightforward to calculate the probability  
 that receiver  $D_{ij}$  recovers  $m$  out of  $k_{ij}$  original packets ( $m <$   
 $k_{ij}$ ). That is

$$P_{ij}(m) = \sum_{l=m}^{K_i-1} \binom{N_i-k_{ij}}{l-m} \binom{k_{ij}}{m} p_{ij}^{N_i-l} (1-p_{ij})^l. \quad (17)$$

Let a random variable  $\gamma_{ij}$  denote the PQoS of receiver  $D_{ij}$ .  
 Then, the expected PQoS over all users is given by

$$\gamma(\mathcal{G}) = \frac{1}{M} \mathbb{E} \left[ \sum_{i=1}^{|\mathcal{G}|} \sum_{j=1}^{M_i} c_{ij} \gamma_{ij} \right]. \quad (18)$$

Therefore, a partition scheme is optimal, if it maximizes the ex-  
 pected PQoS across all users. The CARE optimization problem  
 can be formulated as

$$\text{CARE : } \max_{\mathcal{G} \in \Omega} \left\{ \frac{1}{M} \sum_{i=1}^{|\mathcal{G}|} \sum_{j=1}^{M_i} c_{ij} \mathbb{E}[\gamma_{ij}] \right\}$$

$$\text{s.t. : } \sum_{i=1}^{|\mathcal{G}|} \sum_{j=1}^{M_i} n_{ij} = N \quad (19)$$

$$\sum_{i=1}^{|\mathcal{G}|} \sum_{j=1}^{M_i} k_{ij} = K \quad (20)$$

$$0 \leq c_{ij} \leq 1 \text{ for } i = 1, 2, \dots, |\mathcal{G}| \\ j = 1, 2, \dots, M_i \quad (21)$$

$$\mathbb{E}[\gamma_{ij}] = \mathcal{F}(k_{ij}) P_{ij}^s + \sum_{m=0}^{k_{ij}-1} \mathcal{F}(m) P_{ij}(m) \quad (22)$$

where  $\Omega$  denotes the collection of all nonempty-subset parti-  
 tions of  $M$  flows. The objective is to maximize the average  
 expected PQoS across all the receivers. The first constraint  
 represents the maximum number of time slots available for  
 transmission in a data frame. The quantity  $\sum_{j=1}^{M_i} n_{ij}$  gives the  
 number of time slots allocated for group  $i$ ,  $i = 1, \dots, |\mathcal{G}|$ . The  
 second constraint accounts for the total number of original data  
 packets that need to be transmitted, i.e.,  $\sum_{j=1}^{M_i} k_{ij}$ . Finally, the

583 last two constraints, respectively, express the weight factor and  
584 the equation to compute the expected PQoS based on the type  
585 of applications and the number of received packets for receiver  
586  $j$  in group  $i$ .

587 3) *CARE Hardness*: We next show that finding the optimal  
588 solution to the CARE problem is NP-hard. In particular, we will  
589 show 1) an existing algorithm that reduces an instance of CARE  
590 to an instance of the stochastic reward knapsack problem (SKP)  
591 [20] in polynomial time, and show that, 2) given the solution of  
592 CARE, the solution of the SKP can be found in polynomial time.

593 *SKP Description*: The SKP problem is described as  
594 follows. Given a set of  $n$  items, where item  $i$  has a fixed  
595 weight  $w_i$  and a random value  $v_i$ , the distribution of  $v_i$  could  
596 be unknown. The objective is to select a subset of items such  
597 that it maximizes the sum values while not exceeding the limit  
598 capacity of the knapsack.

599 *Instance Reduction*: We assume that there are  $M$  in-  
600 formation flows, represented by their FEC codes  $(n_i, k_i)$ ,  $i =$   
601  $1, \dots, M$ , that we want to deliver to  $M$  receivers within  $N$  time  
602 slots. Without loss of generality, we assume that  $\sum_{i=1}^M n_i > N$ ,  
603 i.e., only a subset of the  $M$  flows is selected. The expected  
604 PQoS of each subgroup of the  $M$  flows is considered as reward  
605  $v_i$  of an item of the SKP problem. It is clear that  $v_i \in V$   
606 is a random variable, where its value depends on the erasure  
607 probability  $p_i$  and the flow partition. For simplicity, we let the  
608 priority factor  $c_i = 1 \forall i = 1, \dots, M$ . Therefore, we have an  
609 instance of the SKP corresponding to a partition of the  $M$  flows  
610 into subgroups.

611 *Solution Reduction*: Finding the solution of SKP, given  
612 the solution of CARE, is straightforward. Let us assume that  $\mathcal{G}^*$   
613 is a partition that maximizes the expected PQoS across all the  
614 users. We have that

$$\begin{aligned} \sum_{i=1}^{|\mathcal{G}^*|} \sum_{j=1}^{M_i} \mathbb{E}[\gamma_{ij}] &\stackrel{(a)}{\geq} \sum_{i=1}^{|\mathcal{G}|} \sum_{j=1}^{M_i} \mathbb{E}[\gamma_{ij}] \quad \forall \mathcal{G} \in \Omega \\ &\stackrel{(b)}{\geq} \sum_{i=1}^M x_i v_i \quad \forall v_i \in V \\ x_i &= \{0, 1\}, \sum_{i=1}^M x_i w_i \leq N. \end{aligned} \quad (23)$$

615 The inequality (a) follows from the assumption that  $\mathcal{G}^*$  is the  
616 solution to CARE, i.e., the optimal one that maximizes the ex-  
617 pected PQoS. The inequality (b) immediately holds because the  
618 right-hand side of (b) is equivalent to that of (a), representing  
619 in the context of SKP, i.e., the value distribution of the items.  
620 Thus, the union of the subsets of  $\mathcal{G}^*$  forms the subsets of items  
621 selected for the SKP, where the total value that is maximized  
622 with the weight is less than the knapsack capacity.

#### 623 D. Context-Aware Partial Interflow Network Coding

624 We next discuss PCARE, an extension of CARE, to further  
625 improve the system PQoS. We want to emphasize that PCARE  
626 has different formulation in comparison with CARE. On one  
627 hand, CARE seeks for optimal mixing and scheduling of in-  
628 coming data flows, whereby a flow (as a whole) can be either

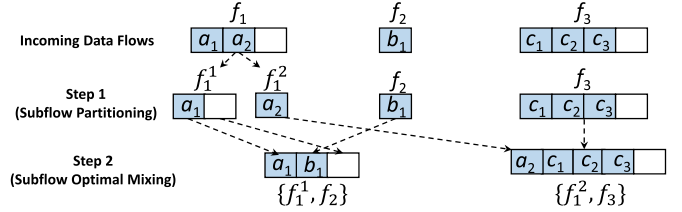


Fig. 4. Example of subflow partitioning and mixing of PCARE.

combined with other flows or transmitted separately. On the  
629 other hand, PCARE allows incoming data flows to be further  
630 divided into subflows, which consequentially are combined  
631 with other subflows (of other flows) for transmission. By doing  
632 so, PCARE can achieve a finer grained bandwidth allocation  
633 compared with CARE, albeit a more sophisticated computation.  
634 Fig. 4 illustrates an example of subflow partitioning and mixing  
635 of PCARE. In this example, we consider three incoming data  
636 flows with their corresponding FEC codes being  $(n_1, k_1) =$   
637  $(3, 2)$ ,  $(n_2, k_2) = (1, 1)$ , and  $(n_3, k_3) = (4, 3)$ , respectively.  
638 We recall that an FEC code  $(n, k)$  implies that  $n$  time slots  
639 are used to deliver  $k$  data packets, where receiving any  $k$  out  
640 of  $n$  transmitted packets is sufficient to recover the data. The  
641 PCARE scheme is performed via the following steps. 642

- *Subflow Partitioning*: PCARE first divides incoming data  
643 flows into subflows. We emphasize that there are many  
644 ways to divide a flow  $f_i : (n_i, k_i)$  into subflows. Let  $\mathbf{f}$   
645 be the set representing all subflows of  $f_i$ . Consider a  
646 configuration where  $f_i$  is divided into  $J$  subflows  $f_i^j$ ,  $j =$   
647  $1, \dots, J$ . We have the following constraints: 648

$$\mathbf{f} \supset f_i^j : (n_i^j, k_i^j), j = 1, \dots, J$$

$$n_i = \sum_{j=1}^J n_i^j, k_i = \sum_{j=1}^J k_i^j.$$

In Fig. 4, flow  $f_1$  is divided into  $J = 2$  subflows  $f_1^1$  :  
649  $(n_1^1, k_1^1) = (2, 1)$  and  $f_1^2$  :  $(n_1^2, k_1^2) = (1, 1)$ , while we  
650 keep flows  $f_2$  and  $f_3$  intact (a flow is a subflow of  
651 itself). As shown, even with this simple example, there are  
652 many ways to divide the incoming flows into subflows.  
653 Our example in Fig. 4 shows only one specific subflow  
654 configuration. 655

- *Subflow Optimal Mixing*: Next, given a set of subflows,  
656 they are then optimally mixed together for transmission.  
657 In our example in Fig. 4, subflow  $f_1^1$  is combined with  
658 flow  $f_2$ , whereas subflow  $f_1^2$  is combined with flow  $f_3$ .  
659 Data packets of each subgroup are then combined together  
660 within that subgroup for transmission. The key idea of  
661 PCARE is to divide the incoming flows into subflows,  
662 enabling finer grained bandwidth allocation for each sub-  
663 group. To illustrate this point, we next show a simple  
664 numerical example to illustrate the benefit of subflow  
665 partitioning of the PCARE scheme. 666

*Numerical Example*: Assume that  $p_1 = 0.06$ ,  $p_2 = 0.2$ , and  
667  $p_3 = 0.05$  are the packet loss rates of the channels to receivers 668



669  $D_1$ ,  $D_2$ , and  $D_3$ , respectively. Supposing that we use RNC  
 670 for combining data within each subgroup, we then have the  
 671 probability that all receivers can recover their desired data using  
 672 the PCARE scheme in Fig. 4, which is computed as follows.

673 • *Receiver  $D_1$* : The probability that receiver  $D_1$  can re-  
 674 cover its desired data is computed by the product of the  
 675 probabilities that data in each subgroup can be recovered  
 676 successfully. In our example in Fig. 4, there are two  
 677 subgroups, which have two and four data packets being  
 678 transmitted in three and five time slots, respectively. The  
 679 probability that  $D_1$  can recover its data is computed as

$$P_1 = \sum_{i=2}^3 \binom{3}{i} (1-p_1)^i p_1^{3-i} \times \sum_{i=4}^5 \binom{5}{i} (1-p_1)^i p_1^{5-i}. \quad (24)$$

680 The first and second terms represent the probability that  
 681  $D_1$  successfully receives data packets  $a_1$  and  $a_2$  in the  
 682 first and second subgroups, respectively.

683 • *Receivers  $D_2$  and  $D_3$* : Similarly, we can compute the  
 684 probabilities that receivers  $D_2$  and  $D_3$  can recover their  
 685 desired data as

$$P_2 = \sum_{i=2}^3 \binom{3}{i} (1-p_2)^i p_2^{3-i} \quad (25)$$

$$P_3 = \sum_{i=4}^5 \binom{5}{i} (1-p_3)^i p_3^{5-i}. \quad (26)$$

686 Substituting the values of  $p_1 = 0.06$ ,  $p_2 = 0.2$ , and  $p_3 =$   
 687  $0.05$  to (24)–(26), we obtain  $P_1 = 0.9581$ ,  $P_2 = 0.8960$ , and  
 688  $P_3 = 0.9974$ . As a result, the probability that all receivers can  
 689 recover their desired data is  $P = P_1 \times P_2 \times P_3 = 0.8391$ .

690 On the other hand, the best solution CARE can be achieved  
 691 by combining flows 1 and 3 together, while transmitting flow 2  
 692 separately. Using this scheme, all receivers can recover their  
 693 desired data with a probability  $P = 0.792$ , which is much  
 694 lower than that of the PCARE scheme earlier. Therefore, we  
 695 can see that, by dividing traffic flows into subflows, PCARE  
 696 achieves better bandwidth allocation, resulting in higher system  
 697 performance. However, one can also see that finding the optimal  
 698 solution to the PCARE scheme is extremely computationally  
 699 expensive. Its detailed theoretical analysis is considered as our  
 700 future investigation.

## 701 V. APPROXIMATION ALGORITHM TO CONTEXT-AWARE 702 INTERFLOW NETWORK CODING AND SCHEDULING

703 Here, we describe a simple but efficient heuristic algorithm  
 704 based on the well-known MCMC method [46] to find a near-  
 705 optimal solution. Although MCMC-based techniques have been  
 706 extensively used to solve hard optimization problems, it is  
 707 very challenging to devise an efficient algorithm to approx-  
 708 imate the solution of a given specific problem. Typically, a  
 709 system designer needs to 1) design the target distribution that  
 710 reflects the solution and 2) effectively generate samples from  
 711 this target distribution. Unfortunately, achieving such design  
 712 goals is challenging because it is very difficult to know the  
 713 stochastic properties of the system. Additionally, it is not trivial

to generate samples from an arbitrary distribution and design  
 714 mechanism to transition among the states. Furthermore, the  
 715 convergence time of the proposed algorithm needs to be upper  
 716 bounded to ensure the performance guarantee. Here, we will  
 717 describe how to obtain such objectives. We start with the  
 718 description of the MCMC-based approximation algorithm and  
 719 then prove its upper bounded convergence time. 720

### A. MCMC-Based Algorithm (CARE-SAB) 721

Here, we show how to appropriately construct a target  
 722 distribution and use MCMC to obtain the solution. Consider a  
 723 scenario with  $M$  concurrent flows traversing through the base  
 724 station. Let  $\Omega$  be the set of all possible partition policies and  
 725  $S(\pi)$  be the average satisfaction factor of a partition policy  
 $\pi \in \Omega$ . We represent each partition policy by an  $M$ -tuple group  
 727 index as  $\pi = (i, j, \dots, k)$ , where  $i$  indicates that the first flow  
 728 belongs to group  $i$ , the second flow belongs to group  $j$ , and  
 729 so on. The objective is to maximize the average PQoS over all  
 730 users. That is 731

$$\max_{\pi \in \Omega} S(\pi) = \max_{\pi \in \Omega} \left\{ \sum_{i=1}^{|\pi|} \sum_{j=1}^{M_i} c_{ij} \gamma_{ij} \right\}. \quad (27)$$

We should note that the size of  $\Omega$ , i.e., the number of ways that  
 732  $M$  flows can be partitioned into subgroups, is very large. Based  
 733 on the result of [47], we have that 734

$$|\Omega| = \sum_{j=1}^M \frac{1}{j!} \sum_{i=0}^j (-1)^i \binom{j}{i} (j-i)^M. \quad (28)$$

Hence, using exhaustive search, even for a reasonably small  
 735 number of flows, is infeasible for time-sensitive applications.  
 736 Moreover, every time a flow joins, terminates, or its channel  
 737 condition changes, the AP needs to repartition again. Instead,  
 738 by using the MCMC method, we will show that the time to  
 739 achieve the near-optimal solution will be substantially reduced. 740

We first define the target distribution as follows: 741

$$f(\pi) = C e^{\frac{S(\pi)}{T_B}} \quad (29)$$

where  $C$  is a normalization factor, and  $T_B$  is a “cooling” param-  
 742 eter that controls the process convergence. In particular, when  
 743  $T_B$  reaches to a sufficient small value, the algorithm terminates,  
 744 and the best accepted configuration is returned as the solution.  
 745 As shown in (29), when  $S(\pi)$  increases,  $f(\pi)$  also increases.  
 746 Therefore, with high probability, we will draw samples  
 747 corresponding to  $S(\pi)$ , which, by design, will maximize the  
 748 average user PQoS. Next, we design a mechanism for moving  
 749 from one state to another in the chain. To do so, we define a  
 750 neighbor of a partition in the sample space  $\Omega$  as follows. 751

*Definition 5.1: A Partition Policy  $\pi_j$  is Called a Neighbor of 752*  
*a Partition Policy  $\pi_i$  iff  $\pi_i$  and  $\pi_j$  Differ in Only One Element: 753*  
 From the aforementioned definition,  $\pi_j$  can be generated from  
 754  $\pi_i$  by replacing an element of  $\pi_i$  with an element drawn  
 755 randomly from the index set  $\mathcal{I} = \{1, 2, \dots, M\}$ . For exam-  
 756 ple, when  $M = 5$ , partition  $\pi_i = (1, 1, 3, 2, 3)$  has a neighbor  
 757  $\pi_j = (1, 1, 1, 2, 3)$  (because  $\pi_i$  and  $\pi_j$  differ only in the  
 758 element). 759

760 One should note that, in the context of the MCMC, each  
761 partition policy corresponds to a state. We now propose a  
762 simulated-annealing-based algorithm to generate samples ac-  
763 cording to the designed target distribution. We propose a tran-  
764 sition function  $q(\pi_i, \pi_j)$  that specifies the probability to move  
765 from state  $\pi_i$  to one of its neighboring states  $\pi_j$ . Specifically,  
766 an element of  $\pi_i$  is selected uniformly at random, and then it is  
767 replaced by one of the possible indexes uniformly. Therefore,  
768 we have that

$$q(\pi_i, \pi_j) = q(\pi_j, \pi_i) = \frac{1}{2M(M-1)}. \quad (30)$$

769 Consequently, the acceptance probability, i.e., the probability  
770 that the chain moves from the current state  $\pi_i$  to a neighboring  
771 state  $\pi_j$ , is given by

$$\begin{aligned} \alpha(\pi_i, \pi_j) &= \min \left\{ 1, \frac{f(\pi_j)q(\pi_j, \pi_i)}{f(\pi_i)q(\pi_i, \pi_j)} \right\} \\ &= \begin{cases} 1, & \text{if } S(\pi_j) \geq S(\pi_i) \\ e^{\frac{S(\pi_j) - S(\pi_i)}{T_B}}, & \text{if } S(\pi_j) < S(\pi_i). \end{cases} \end{aligned} \quad (31)$$

772 As defined, the chain will move from the current state  $\pi_i$  to a  
773 new state  $\pi_j$  with probability of one, if  $\pi_j$  is a better state (state  
774 has higher user satisfaction value). Otherwise, the chain will  
775 move to a new state  $\pi_j$  with probability of  $e^{(S(\pi_j) - S(\pi_i))/T_B}$ .  
776 With this design, the whole state space will be explored,  
777 if we run the algorithm sufficiently long. Furthermore, the  
778 Boltzmann distribution will become increasingly more concen-  
779 trated around the global maximizer, by gradually decreasing the  
780 temperature  $T_B$ . Pseudocode of the simulated-annealing-based  
781 algorithm is described in Algorithm 1.

---

### Algorithm 1: CARE-SAB Algorithm.

---

782 **Input:**  $M, c_i$ , FEC code  $(n_i, k_i)$ .  
783 **Output:** Optimal Flow Partition.  
784 1: **STEP 1:** Initialize the starting state  $\pi_0$  and temperature  
785  $T_0$ . Set  $n = 0$ .  
786 2: **STEP 2:** With probability 1/2, generate a new state  $\pi_j$   
787 from the proposal  $q(\pi_n, \pi_j)$ .  
788 3: **STEP 3:**  
789 4: **if**  $S(\pi_j) \geq S(\pi_n)$  **then**  
790 5:  $\pi_{n+1} = \pi_j$   
791 6: **else**  
792 7:  $U \sim U(0, 1)$  {Generate a uniform random variable.}  
793 8: **if**  $U < \alpha(\pi_n, \pi_j) = e^{\frac{S(\pi_j) - S(\pi_n)}{T_n}}$  **then**  
794 9:  $\pi_{n+1} = \pi_j$   
795 10: **else**  
796 11:  $\pi_{n+1} = \pi_n$   
797 12: **end if**  
798 13: **end if**  
799 14: **STEP 4:** Decrease the temperature  $T_{n+1} = \beta T_n$  where  
800  $\beta < 1$ , increase  $n$  by 1 and repeat from **STEP 2** until  
801 stopping condition satisfied (\*).  
802 15: **STEP 5:** Return a scheme  $\pi$  that produces the  
803 maximum weighted-average satisfaction factor.

---

*Remark (\*):* The stopping condition can be set by the number 804  
of iterations or the difference between the two consecutive 805  
states is less than a prespecified value. 806

### B. Upper Bound of Convergence Time 807

1) *Convergence Correctness:* The guarantee of convergence 808  
to the target distribution using the CARE-SAB algorithm is 809  
shown via the following theorem. 810

*Theorem 5.2:* Samples drawn from the CARE-SAB algo- 811  
rithm form a Markov chain (MC) whose states satisfy the 812 AQB  
detailed balance equation 813

$$\theta(\pi_i)P(\pi_i, \pi_j) = \theta(\pi_j)P(\pi_j, \pi_i) \quad \forall \pi_i, \pi_j \in \Omega \quad (32)$$

where  $\theta(\pi_i)$  and  $\theta(\pi_j)$  are the stationary distributions of states 814  
 $\pi_i$  and  $\pi_j$ ;  $P(\pi_i, \pi_j)$  and  $P(\pi_j, \pi_i)$  are, respectively, the transi- 815  
tion probabilities from state  $\pi_i$  to state  $\pi_j$  and vice versa. 816

*Proof:* The proof is provided in the Appendix. ■ 817

The result of Theorem 5.2 shows that samples drawn from 818  
the designed algorithm form an ergodic MC, i.e., every state can 819  
go to every state; thus, it is possible to find an optimal solution 820  
as long as the algorithm runs sufficiently long. 821

2) *Convergence Time:* Here, we will derive an upper bound 822  
of convergence time to the target distribution. We first define 823  
the variation distance at time step  $k$  with respect to an initial 824  
state  $\pi_0$  of the MC as 825

$$\Delta_{\pi_0}(k) \triangleq \max_{\pi \in \Omega} |P^k(\pi_0, \pi) - \theta(\pi)|. \quad (33)$$

Then, the convergence time of the MC to the target distribution 826  
is measured by 827

$$\tau_{\pi_0}(\varepsilon) \triangleq \min \{k : \Delta_{\pi_0}(k') \leq \varepsilon \quad \forall k' \geq k\} \quad (34)$$

where  $0 < \varepsilon$  is an arbitrary infinitesimal value specifying 828  
how close the desired solution to an optimal solution. To 829  
bound the convergence time at which the chain approaches its 830  
stationary distribution (optimal solution), we use the canon- 831  
ical path technique [21]. Letting  $e = (\pi_x, \pi_y) \in \Omega^2$ , we de- 832  
fine  $Q(e) \triangleq Q(\pi_x, \pi_y) = \theta(\pi_x)P(\pi_x, \pi_y)$ , and a graph  $G =$  833  
 $(\Omega, E)$ , where  $(\pi_x, \pi_y) \in E$  iff  $Q(\pi_x, \pi_y) > 0$ . For every or- 834  
dered pair  $(\pi_x, \pi_y) \in \Omega^2$ , a canonical path  $\varsigma_{xy}$  through  $G$  from 835  
 $\pi_x$  to  $\pi_y$  is specified by a sequence of legal transitions in  $G$  836  
that leads from initial state  $\pi_x$  to final state  $\pi_y$ . Let  $\Gamma \triangleq \{\varsigma_{xy} : 837$   
 $\pi_x, \pi_y \in \Omega\}$  denote the set of all canonical paths. We now 838  
define the edge congestion for the set  $\Gamma$  as follows: 839

$$\rho(\Gamma) \triangleq \max_{e \in E} \frac{1}{Q(e)} \sum_{\varsigma_{xy} \ni e} \theta(\pi_x)\theta(\pi_y)|\varsigma_{xy}| \quad (35)$$

where  $\gamma_{xy} \ni e$  implies that  $\varsigma_{xy}$  uses the directed edge  $e$ , and 840  
 $|\varsigma_{xy}|$  denotes the length of the path. The convergence time is 841  
bounded by the following proposition. 842

*Proposition 5.1:* Let  $\mathbf{M}$  be a finite, time-reversible, and 843  
ergodic MC over  $\Omega$  with self-loop probabilities  $P(x, x) \geq 1/2$  844  
for all  $x \in \Omega$  and stationary distribution  $\pi$ . If the congestion 845  
of  $\mathbf{M}$  is  $\rho$ , then the mixing time of  $\mathbf{M}$  satisfies  $\tau_x(\varepsilon) \leq 846$   
 $\rho(\ln \pi(x)^{-1} + \ln \varepsilon^{-1})$ , for any choice of initial state  $x$ . 847

848 We are now ready to bound the mixing time of the proposed  
 849 approximation algorithm CARE-SAB. We choose canonical  
 850 paths  $\varsigma_{xy}$  from any state  $\pi_x$  to any state  $\pi_y$ , which takes  $T$  steps,  
 851 changing  $\pi_{x_i}$  to  $\pi_{y_i}$  on the  $i$ th step. Thus, when  $\pi_{x_i} = \pi_{y_i}$  for  
 852 some  $i$ , the  $i$ th step is just a self-loop. We derive a bound for the  
 853 mixing time of the chain. Considering any edge  $e = (\pi_u, \pi_v)$   
 854 for  $u \neq v$ , we have

$$\begin{aligned} Q(e) &= \theta(\pi_u)P(\pi_u, \pi_v) \stackrel{(a)}{=} C e^{\frac{S(\pi_u)}{T_B}} \frac{\min \left\{ 1, e^{\frac{S(\pi_u) - S(\pi_v)}{T_B}} \right\}}{2M(M-1)} \\ &= \frac{C}{2M(M-1)} \min \left\{ e^{\frac{S(\pi_u)}{T_B}}, e^{\frac{S(\pi_v)}{T_B}} \right\} \stackrel{(b)}{\geq} \frac{C}{2M(M-1)} \end{aligned} \quad (36)$$

855 where (a) follows from (29)–(31), and (b) follows by using the  
 856 fact that a partition policy could have a zero-PQoS value. In  
 857 addition, we have

$$\theta(\pi_u)\theta(\pi_v) = C^2 e^{\frac{S(\pi_u) + S(\pi_v)}{T_B}} \leq C^2 e^{\frac{2}{T_B}} \quad (37)$$

858 where the last inequality follows from the fact that  $S(\pi) \leq 1$   
 859 for any partition scheme  $\pi$ . We now compute the number of  
 860 canonical paths  $\varsigma_{xy}$  that use edge  $e$ . Note that a canonical path  
 861  $\varsigma_{xy}$  uses edge  $e = (\pi_u, \pi_v)$ , where  $\pi_u$  and  $\pi_v$  differ only in the  
 862  $i$ th element iff  $\pi_{x_j} = \pi_{u_j}$  for  $j = i, \dots, T$  and  $y_j = v_j$  for  $j =$   
 863  $0, \dots, i$ . Thus, the number of canonical paths that use edge  $e$  is  
 864  $K_m^{T-1}$ . We now bound the edge congestion as

$$\begin{aligned} \rho &= \max_{e \in \Omega} \frac{1}{Q(e)} \sum_{\varsigma_{xy} \ni e} \pi(x)\pi(y) |\varsigma_{xy}| \\ &\leq 2CT^2 (K_m - 1)(K_m)^{T-1} e^{\frac{2NK_m}{T_B}}. \end{aligned} \quad (38)$$

865 In addition, we have  $C = 1 / \sum_{x \in \Omega} e^{(S(x))/T_B} \leq 1/|\Omega| =$   
 866  $1/K_m^T$ . Therefore

$$\rho \leq 2T^2 e^{\frac{2NK_m}{T_B}}. \quad (39)$$

867 Using the result from Proposition 5.1 and noting that  $\pi(x) \geq$   
 868  $1/|\Omega| = 1/K_m^T$ , we then have the convergence time bounded by

$$\tau_x(\epsilon) \leq 2T^2 e^{\frac{2NK_m}{T_B}} (\ln K_m^T + \ln \epsilon^{-1}). \quad (40)$$

869 As expected, when the value of  $\epsilon$  decreases (i.e., closer to the  
 870 optimal solution), it requires longer runtime. However, as we will  
 871 show in the simulation, on the order of hundred iterations, the  
 872 approximation algorithm can obtain a close-optimal solution.

## 873 VI. SIMULATIONS AND DISCUSSIONS

### 874 A. Basic Setup

875 *Network Parameters:* We consider a realistic wireless  
 876 access network having different types of applications with time-  
 877 varying channel conditions. We first consider a network consist-  
 878 ing of five data flows of different applications and compare the  
 879 performance of different transmission strategies. We then in-  
 880 crease the number of data flows to evaluate the robustness of the  
 881 approximation algorithm. We assume that there are two classes  
 882 of services: delay-sensitive and elastic traffic. The transmitter

TABLE I  
PARAMETERS OF THE INCOMING FLOWS

Flow ID	Rx ID	FEC $(n_i, k_i)$	$p_i$	Service type	Priority
$f_1$	$D_1$	(21, 18)	–	Elastic	1
$f_2$	$D_2$	(21, 18)	0.05	Elastic	1
$f_3$	$D_3$	(22, 18)	0.05	Elastic	2
$f_4$	$D_4$	(31, 25)	0.05	Delay-sensitive	3
$f_5$	$D_5$	(38, 30)	0.05	Delay-sensitive	4

883 decides a coding scheme for a flow, based on the cost at which  
 884 the user had paid to the service provider, i.e., the higher cost, the  
 885 higher priority. Note that the service class and priority of a data  
 886 flow can be easily elaborated in the header of the transmitted  
 887 packets. In the UNI technique, the transmitter uses the priorities  
 888 of the incoming flows to assign their redundancies, and they  
 889 will be used in all the techniques for a fair comparison. We  
 890 consider a wireless channel with a bandwidth of 2 Mb/s, which  
 891 is equivalent to  $N = 133$  time slots or 133 1.5-kB packets. In  
 892 addition, an elastic traffic requires 18 data packets per second,  
 893 corresponding to a rate of 27 kb/s, while a delay-sensitive traffic  
 894 requires 25 and 30 data packets, corresponding to rates of 37.5  
 895 and 45 kb/s, for medium and high QoS, to achieve a PQoS  
 896 value of one. Our parameters are set based on the number of  
 897 frames per second in video streaming and the standard service  
 898 specifications [48]. For example, our delay-sensitive flow can  
 899 be used to model a Voice over IP (VOIP) call (e.g., using G.728  
 900 standard with a codec interval of 5 (ms) requires a transmission  
 901 rate of 31.5 kb/s [48]).

902 If the number of data packets received at each receiver is  
 903 less than the required packets, its satisfaction will decrease in  
 904 accordance to the PQoS functions, as described in Section III-C.  
 905 The transmission parameters of the incoming flows are given  
 906 in Table I. These parameters are set based on the types of ap-  
 907 plications, priorities of the incoming flows, and the bandwidth  
 908 availability. In addition, the redundancy used for each incoming  
 909 flow depends on its priority; for example, in our experiments,  
 910 we set priorities 1, 2, 3, and 4, corresponding to redundancies  
 911 of 15%, 20%, 25%, and 30%, respectively. Note that these  
 912 parameters will be applied to all techniques.

913 *Transmission Strategies:* We evaluate the performance of the  
 914 following transmission strategies for comparison.

- *Unicast (UNI):* The UNI scheme transmits data of the 915 incoming flows separately, without using NC. 916
- *Random Network Coding (RNC):* RNC scheme imple- 917 ments a standard interflow NC technique, where data 918 packets of all flows are combined together using RNC 919 [41]. The coded packets are then broadcasted to all the 920 receivers. 921
- *Systematic Random Network Coding (SRNC):* SRNC im- 922 plements a simple systematic NC, where original data 923 packets are transmitted in the first phase and coded data 924 packets (generated by using RNC across all data flows) 925 are transmitted in the second phase. 926
- *Type of Flow (ToF):* The first naïve ToF scheme mixes 927 all data of flows with the same application types. Such a 928 transmission strategy does not have to perform intensive 929 optimization computation; however, it might limit the 930 effect of mismatched mixing of the NC-based schemes, 931 by combining data of flows with the same types. 932

- 933 • *Priority of Flow (PoF) Scheme*: Differently, the PoF  
 934 scheme selects flows that have the same priority as sub-  
 935 groups to perform NC. The idea of using the PoF scheme  
 936 is to utilize the priority factors of the incoming data flows  
 937 to quickly classify them into different subgroups for NC.  
 938 Intuitively, encoding data of the same priority flows could  
 939 enhance the QoS of the network.
- 940 • *Network Coding for Throughput (NCT)*: The NCT scheme  
 941 is simulated based on the work in [5], which is proposed  
 942 for maximizing the network throughput. In NCT, the  
 943 sender considers the packet at the head of its queue as  
 944 a primary packet and selects side packets to construct  
 945 coded packets so that it maximizes the number of possible  
 946 receivers at the current time slot.
- 947 • *CARE/CARE-EXH*: This scheme uses the optimal mixing  
 948 solution via exhaustive search for data transmission.
- 949 • *CARE-SAB*: CARE-SAB uses an approximation algo-  
 950 rithm based on the proposed CARE-SAB algorithm for  
 951 data encoding and scheduling.
- 952 • *2-PCARE*: Based on the PCARE scheme described in  
 953 Section IV-D, we simulate 2-PCARE for comparison.  
 954 In this scheme, each incoming flow is divided into two  
 955 equal subflows, and then, these subflows are optimally  
 956 combined for transmission.

### 957 B. Data Recovery

958 We first show the benefit of *informed* mixing and the draw-  
 959 back of blind mixing by examining the probability that all the  
 960 receivers can decode their packets using strategies derived in  
 961 Section IV. Fig. 5(a) shows the probability of data recovery  
 962 versus partition policies, i.e., the way of mixing data when  
 963 packet losses of receivers from  $D_2$  to  $D_5$  are set to 5% while  
 964 that of receiver  $D_1$  is 13%. We map each partition policy to  
 965 an integer on the  $x$ -axis. The number of possible partition  
 966 policies is an exponential function of  $M$ , and this is equal to  
 967 the sum of the Stirling numbers of the second kind, as shown in  
 968 (28). We also plot the recoverability probabilities for UNI and  
 969 SRNC techniques on the same graph for comparison. They are  
 970 indicated by straight lines since these techniques do not depend  
 971 on the partition policies. Recall that UNI does not mix packets  
 972 from different flows. SRNC sends the original packets and then  
 973 the mixed redundant packets; hence, the amount of mixing here  
 974 is rather minimal. As observed, SRNC is clearly better than  
 975 UNI. It is interesting to note that, at least in this scenario, blind  
 976 mixing is generally better than UNI. As expected, CARE-SAB  
 977 results in different recoverability probabilities, depending on  
 978 which flows are combined with each other. In our program, we  
 979 let the CARE-SAB algorithm run until converged. Based on our  
 980 deeper data analysis generated by the program, the proposed  
 981 CARE-SAB finds the best partition by mixing flows  $f_1$  and  $f_4$   
 982 into one group and flows  $f_2$ ,  $f_3$ , and  $f_5$  into another group.

983 Next, we evaluate the data recovery probability by all re-  
 984 ceivers versus the packet loss probability in Fig. 5(b). In this  
 985 scenario, the packet loss probabilities of receivers  $D_i, i =$   
 986  $\{2, \dots, 5\}$  are shown in Table I, while the packet loss rate of  
 987 receiver  $D_1$  is varied from 1% to 22%. As expected, CARE-  
 988 SAB outperforms the other schemes, with considerable gaps,

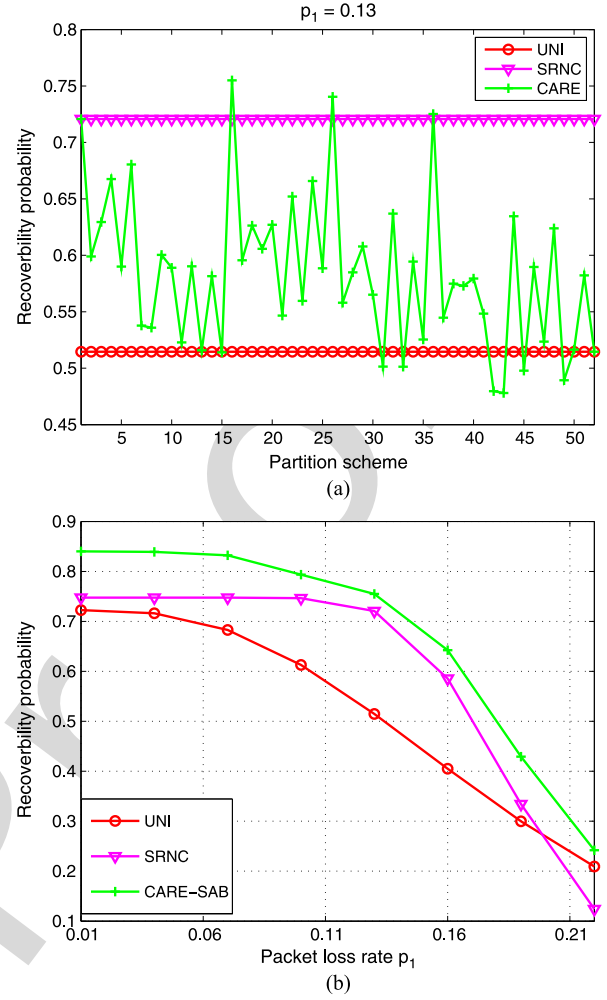


Fig. 5. Recoverability probability versus (a) partition scheme and (b) packet loss rate  $p_1$ .

due to its selective mixing of the flows. In addition, we ob- 989  
 990 serve that, when  $p_1$  is less than 18%, SRNC achieves better  
 991 performance than UNI. In other words, in this erasure regime,  
 992 mixing data packets across all flows would be more beneficial  
 993 than transmitting them separately. However, this is not the case  
 994 when the packet loss  $p_1$  is greater than 18%. In such a setting,  
 995 the UNI scheme outperforms SRNC. The intuition is that SRNC  
 996 combines the data of all the flows; as a result, each receiver  
 997 needs to obtain at least a full set of coded packets to recover its  
 998 data. However, this may not be possible to receiver  $D_1$  because  
 999 it experiences a deep fading, leading to substantial reduction  
 1000 on the overall network recoverability. In such a case, separately  
 1001 transmitting data to different users will be a better option.

### 1002 C. User's PQoS Versus Erasure Probability

We first evaluate the individual PQoS versus the transmission 1003  
 1004 channel conditions. In particular, we set  $p_3 = p_4 = 5\%$ ,  $p_2 =$   
 1005  $p_1 + 0.01$ , and  $p_5 = p_1 + 0.02$ . The other parameters of the  
 1006 network are set the same as before in Table I. The base values  
 1007 of the PQoS, i.e.,  $\gamma_0$ , of the delay-sensitive and elastic traffic  
 1008 are set at 0.5 and 0.6, when the number of useful packets  
 1009 received equals 50% of the intended packets. This setting is to  
 1010 reflect that delay-sensitive applications are more vulnerable to 1010 AQ4

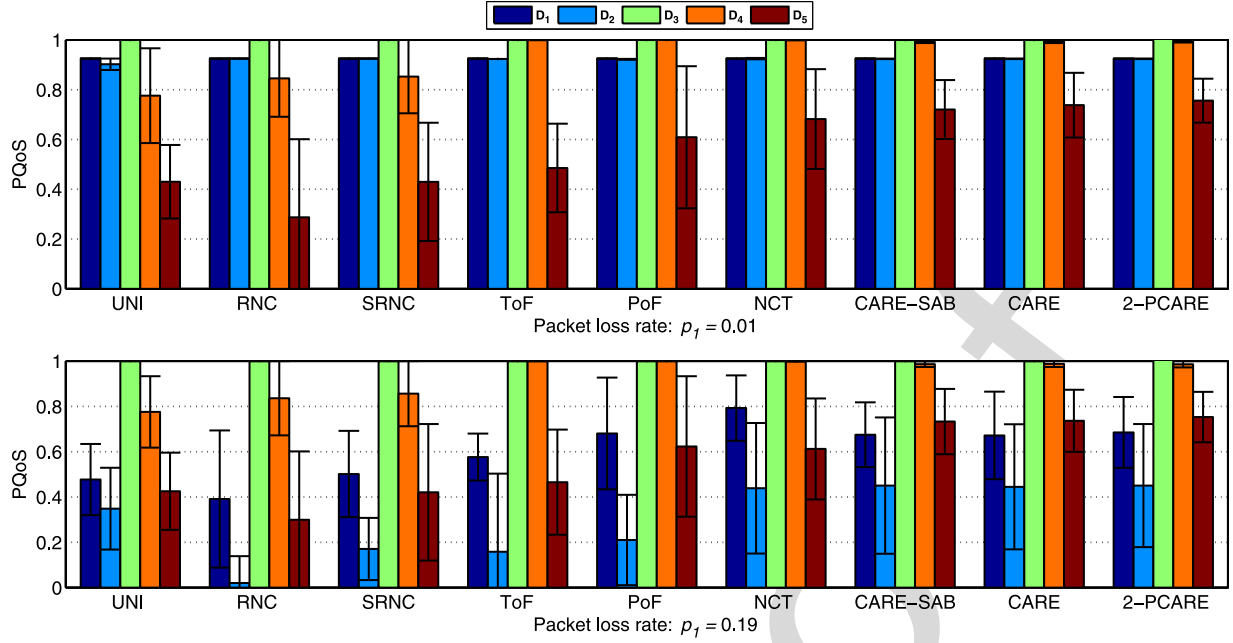


Fig. 6. PqoS values of users in different packet error regimes (order of the receivers is the same, as illustrated in the legend).

1011 packet losses than the elastic traffic. The CARE-SAB algorithm  
 1012 runs for 30 iterations, using the normalization factor  $C = 1$ ,  
 1013 initial temperature  $T_0 = 1$ , and cooling scale factor  $\beta = 0.9$ .  
 1014 We use the simplest partition policy  $k = 2$  for the PCARE  
 1015 scheme, where each flow is evenly divided into two subflows.  
 1016 We perform many trials and compute the average of the results.  
 1017 Fig. 6 represents the PqoS values across all users in two  
 1018 different regimes of the packet erasure probabilities. In the low-  
 1019 packet-loss regime (upper part in Fig. 6), i.e.,  $p_1 = 0.01$ , all  
 1020 the strategies satisfy the QoS of the first four flows. This is  
 1021 intuitively plausible since, in this case, transmission bandwidth  
 1022 is plenty for these flows, no optimization is needed, and all  
 1023 these users get what they want. However, the RNC scheme  
 1024 significantly decreases the PqoS of  $D_5$ . This is because, when  
 1025 combining all the data packets together, it cannot recover the  
 1026 transmitted data, resulting in a degraded PqoS. On the other  
 1027 hand, the CARE-based schemes that balance data types and  
 1028 priorities for different groups achieve the best performance.  
 1029 Furthermore, the 2-PCARE scheme with a finer grained data  
 1030 partition policy achieves the best performance.

1031 On the other hand, in the high-packet-loss regime (lower  
 1032 part in Fig. 6), i.e.,  $p_1 = 0.19$ , all the receivers with low  
 1033 packet loss rates, i.e.,  $D_3$  and  $D_4$ , can still maintain a high  
 1034 PqoS in all transmission strategies. However, the PqoS of the  
 1035 receivers with higher packet loss rates, i.e.,  $D_1$ ,  $D_2$ , and  $D_5$ ,  
 1036 significantly decreases in all strategies. In particular, receiver  
 1037  $D_2$  has its PqoS decreased substantially in the SRNC scheme.  
 1038 The intuition is that mixing up data of all flows makes it  
 1039 difficult for  $D_2$  to receive a full set of the coded packets in  
 1040 the high-erasure-probability regime. As a result,  $D_2$  is not  
 1041 able to recover its data. As expected, CARE (i.e., exhaustive  
 1042 search) that searches all the possibilities of partition policies  
 1043 always achieves the best performance. However, an interesting  
 1044 observation is that the CARE-SAB algorithm can approximate  
 1045 the optimal solution with only 30 iterations. This significantly

reduces the search time compared with the exhaustive search  
 1046 in larger size problems. Indeed, the PqoS achieved by CARE-  
 1047 SAB is very close to that of the exhaustive search CARE, with  
 1048 a marginal gap. 1049

#### D. Network PqoS and Effective Throughput Versus Erasure Probability

1050  
 1051  
 1052 Next, we compare the network PqoS and effective through-  
 1053 put of different transmission strategies versus the packet loss  
 1054  $p_1$ . The effective throughput is computed based only on the  
 1055 received data, contributing to the QoS of the users, without con-  
 1056 sidering application types. Fig. 7(a) shows the average PqoS  
 1057 across all receivers. As expected, the average PqoS decreases  
 1058 with the increase of  $p_1$ . We observe that RNC has the worst  
 1059 PqoS out of all strategies. This is because of the degraded PqoS  
 1060 of receivers that cannot recover the transmitted data due to bad  
 1061 channel conditions. Interestingly, UNI outperforms RNC with a  
 1062 significant performance gap, due to its partially recovered data.  
 1063 On the other hand, in RNC, the receivers need to receive a full  
 1064 set of coded packets for data decoding; however, this may not  
 1065 be possible due to poor transmission channel conditions. The  
 1066 NCT outperforms the other schemes but less than that of the  
 1067 CARE-based schemes, as shown in Fig. 7(a). This is because  
 1068 greedily optimizing the throughput without considering the  
 1069 characteristics of the traffic could significantly decrease the  
 1070 network QoS. Again, 2-PCARE achieves the best performance,  
 1071 which is followed by CARE-based schemes. We further ob-  
 1072 serve that CARE-SAB achieves an identical performance of  
 1073 CARE (i.e., exhaustive search) with much less runtime. This  
 1074 confirms the efficiency and robustness of the proposed CARE-  
 1075 SAB algorithm. 1075

Next, we investigate the network effective throughput versus  
 the packet error rate in Fig. 7(b). As expected, NCT achieves  
 the best performance because its objective is to maximize the



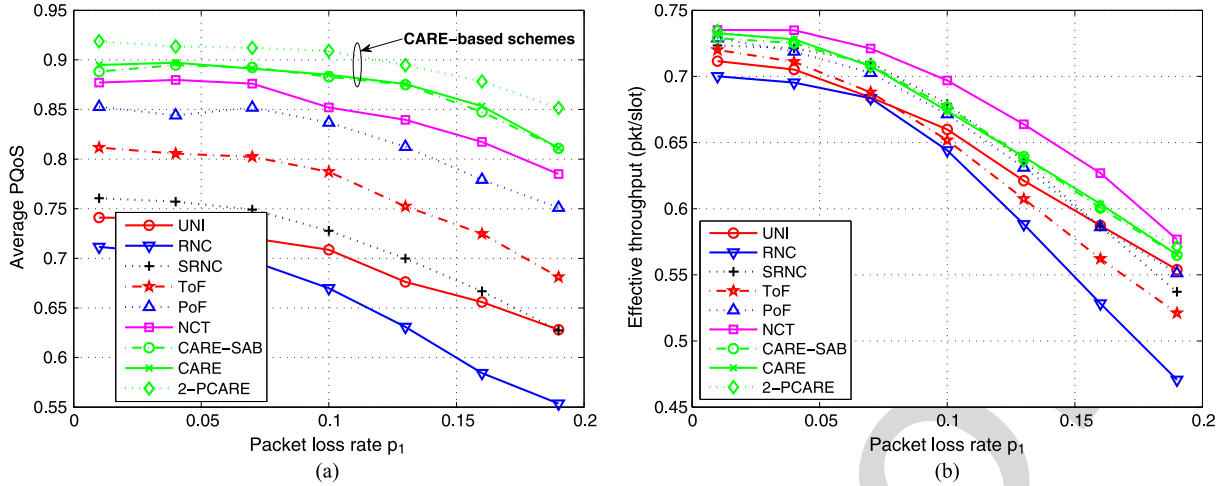


Fig. 7. Network performance of different schemes versus  $p_1$  with  $M = 5$ : (a) average PQoS and (b) effective network throughput.

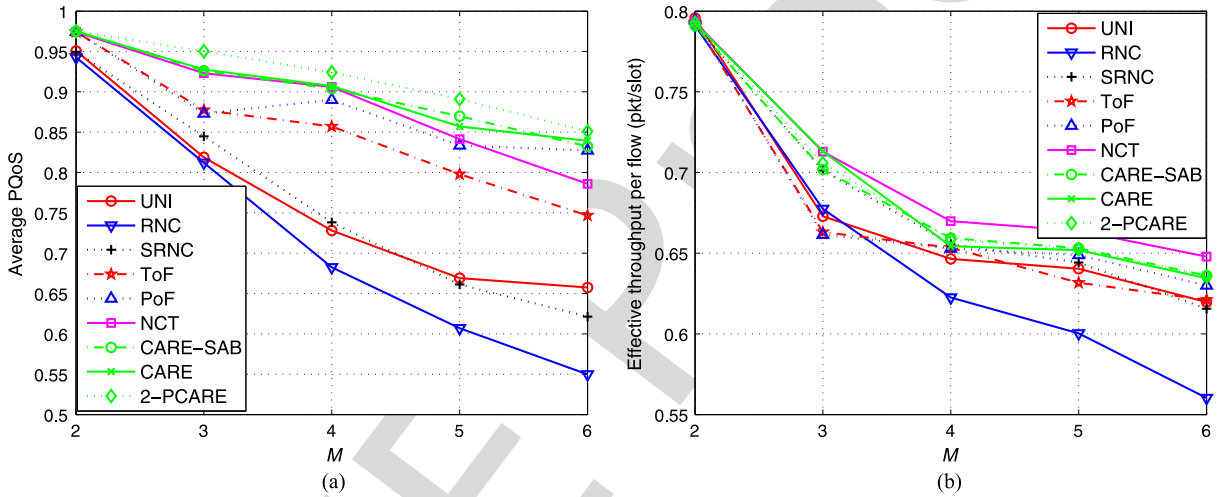


Fig. 8. Network performance of different schemes versus  $M$  for (a) average PQoS and (b) effective network throughput.

1079 network throughput. Interestingly, CARE-based schemes not  
 1080 only outperform the other techniques but also approximate to  
 1081 NCT, despite that the objective of CARE is to maximize the  
 1082 average PQoS. The explanation is as follows. First, CARE  
 1083 uses PQoS in its formulation, whereby the value of PQoS is  
 1084 computed based on both the effective throughput (i.e., num-  
 1085 ber of useful packets) and characteristics of the information  
 1086 flows. Second, and more importantly, PQoS functions are well  
 1087 designed, so that a small change in effective throughput is  
 1088 transformed into the users' PQoS. Thus, optimizing the PQoS  
 1089 results in a near-optimum network effective throughput. We  
 1090 additionally observe that the RNC scheme suffers from com-  
 1091 bining the data of all flows, resulting in the worst performance,  
 1092 particularly in the presence of deep channel fading.

#### 1093 E. PQoS and Throughput Versus Number of Flows

1094 We next examine the overall network PQoS and effective  
 1095 throughput versus the number of data flows in Fig. 8(a) and (b).  
 1096 As expected, the CARE-based schemes outperform the others  
 1097 with considerable gaps. The RNC scheme achieves the worst  
 1098 performance and significantly decreases as  $M$  increases. This

is because all-flow encoding suffers from the curse of “all or  
 nothing” of the RNC decoding constraint, i.e., it requires a full  
 set of encoded packets for data recovery. SRNC outperforms  
 UNI in the regime of lower packet error rate, while significantly  
 decreasing with the increase of  $p_1$ . This is because encoded  
 packets cannot be recovered due to high lost packets. Interest-  
 1104 ingly, the two heuristic ToF and PoF schemes achieve better  
 1105 performance compared with the traditional NC-based and UNI  
 1106 schemes. However, when  $M$  increases, their performance starts  
 1107 decreasing considerably. As expected, the 2-PCARE scheme  
 1108 achieves the best performance, due to its finer grained subflow  
 1109 partition and encoding. We also observe that the CARE-SAB  
 1110 obtains a competitive performance by using only 30 iterations.

1111 Additionally, Fig. 8(b) illustrates the network effective  
 1112 throughput versus  $M$ . As expected, the NCT that is optimized  
 1113 for the throughput achieves the best performance. CARE-based  
 1114 schemes outperform the other techniques, despite that the ob-  
 1115 jective of CARE is to maximize PQoS. This is because PQoS  
 1116 is constructed from both the effective throughput (i.e., number  
 1117 of useful packets) and characteristics of the information flows.  
 1118 Thus, optimizing the PQoS will result in high network effective  
 1119 throughput. The RNC achieves the worst network throughput, 1120

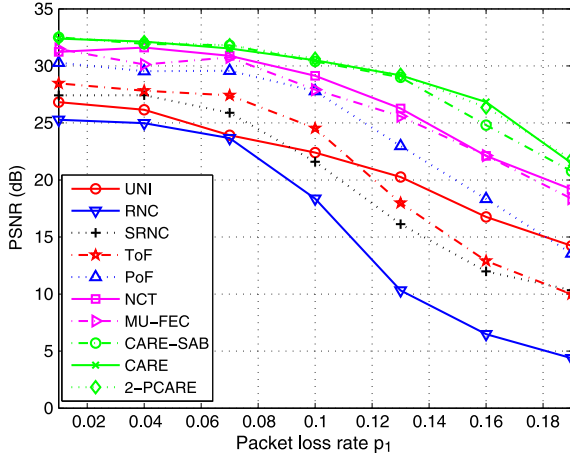


Fig. 9. Average PSNR of video sequences at receivers versus packet lost rate  $p_1$ .

1121 and its performance decreases significantly with the increase of  
 1122  $M$ . This is consistent with the intuition that when  $M$  increases  
 1123 it is more difficult for the receivers to receive a full set of  
 1124 encoded data for decoding.

1125 F. PSNR Versus Erasure Probability

1126 In addition, we also evaluate the average PSNR based on the  
 1127 luminance (Y) component of *Foreman* and *Coastguard* video  
 1128 sequences. We assume that each frame of the video sequences  
 1129 is packetized into an independent network abstraction layer of  
 1130 1500 bytes. Furthermore, in our simulation, we use short video  
 1131 sequences, with the *Foreman* and *Coastguard* having 30 and  
 1132 25 packets, respectively. The longer video sequences can be  
 1133 constructed by concatenating multiple frames. Additionally, we  
 1134 assume that the PSNR of the encoded sequences *Foreman* and  
 1135 *Coastguard* are 29.95 dB [15] and 45.72 dB [6], respectively,  
 1136 corresponding to no-error transmission of those streams.

1137 Fig. 9 illustrates the average PSNR of the two video se-  
 1138 quences using different transmission strategies. In our simu-  
 1139 lation, the transmitter transmits five data flows, consisting of  
 1140 three elastic and two delay-sensitive flows, to five receivers.  
 1141 To compute PSNR, we extract only the received packets of  
 1142 the two video sequences. As expected, when the packet lost  
 1143 rate is small, all schemes perform well with high PSNR. This  
 1144 is because only some of the transmitted data packets were  
 1145 lost. However, when the packet lost rate increases, the video  
 1146 quality at the receivers rapidly degrades. As expected, the  
 1147 proposed CARE-based strategies achieve the highest video  
 1148 qualities, due to their content-aware encoding and scheduling.  
 1149 We also simulated the MU-FEC scheme proposed in [49] for  
 1150 comparison. The MU-FEC scheme exploits intra- and interflow  
 1151 NC for mixing data of different flows at the transmitter to  
 1152 improve bandwidth efficiency. In spite of optimizing its coding  
 1153 for maximizing the network throughput, it does not consider the  
 1154 content of the traffic, resulting in low PSNR. This is the same  
 1155 observation for the NCT scheme, which focuses on through-  
 1156 put maximization instead of network QoS. Interestingly, we  
 1157 observe that the heuristic PoF scheme achieves very good  
 1158 performance in terms of PSNR by combining data of only

higher priority flows associated with the video sequences. In  
 1159 the regime of high erasure rate, with a smaller encoding data  
 1160 batch, it can successfully transmit data to the receivers with  
 1161 higher probability. On the other hand, RNC combining all  
 1162 data together for transmission suffers because most of the data  
 1163 cannot be recovered at the receivers.  
 1164

G. CARE-SAB Convergence Rate

1165 We now evaluate the effectiveness and robustness of the  
 1166 CARE-SAB algorithm. In these experiments, we consider data  
 1167 traffic consisting of eight data flows, where the first five flows  
 1168 are the same as before (in Table I), and the additional flows  
 1169 include one elastic and two delay-sensitive flows that are a  
 1170 copy of  $D_2$  and  $D_4$ , respectively. We have a total of more  
 1171 than  $4 \times 10^3$  possible flow partition policies. To illustrate the  
 1172 convergence of the proposed CARE-SAB, we vary the channel  
 1173 conditions randomly with the packet loss rates in the range  
 1174 between 1% and 20%. In the CARE-SAB algorithm, the initial  
 1175 state is set by grouping all flows together. Fig. 10(a) and (b) il-  
 1176 lustrates a snapshot of the PQoS values changing with respect to  
 1177 the iteration and actual runtime, respectively. The CARE-EXH  
 1178 achieves the best performance by using exhaustive search for  
 1179 all possible partition policies and selects the best partition. On  
 1180 the other hand, CARE-SAB schemes implement the proposed  
 1181 approximation method to find the optimal partition policy. The  
 1182 CARE-SAB-Iter represents state by state the chain visits in  
 1183 each iteration, whereas the CARE-SAB records the maximum  
 1184 PQoS values, which have been obtained up to that iteration.  
 1185 The CARE-SAB schemes first aggressively “explore” states,  
 1186 even the ones with low value of PQoS, and gradually “cool”  
 1187 down to the optimal solution. As illustrated in Fig. 10(a), it  
 1188 takes about 200 iterations (about 5% of the search space) for the  
 1189 CARE-SAB schemes to “hit” the optimal solution, indicated  
 1190 by the PQoS merged to that of the CARE-EXH. The simu-  
 1191 lation result is consistent with the theoretical analysis of the  
 1192 Theorem 5.2, i.e., setting the number of iterations sufficiently  
 1193 large, an arbitrarily near-optimal solution can be obtained via  
 1194 the proposed CARE-SAB algorithm.  
 1195

1196 We further measure the actual runtime of the algorithm based  
 1197 on the elapsed time of the algorithm implemented on our  
 1198 laptop (OS Window 7, Intel Core i5 with 4-GB RAM). As illus-  
 1199 trated in Fig. 10(b), it takes about 0.46 s for the CARE-SAB  
 1200 schemes to find the optimal solution for the case of  $M=8$   
 1201 flows. We note that this is only one snapshot of a trial. In prac-  
 1202 tice, the runtime could be much less than that, if we eliminate  
 1203 the effect of other concurrent processes in the machine.

1204 We further evaluate the convergence robustness of the pro-  
 1205 posed CARE-SAB by using different values of  $\beta$ . We recall  
 1206 that  $\beta$  is the parameter controlling the “cooling” process of  
 1207 the search algorithm. Fig. 11(a) illustrates the convergence of  
 1208 CARE-SAB-Iter for different values of  $\beta$  in  $[0.9, 0.99]$ . As  
 1209 observed, it requires more time for the system with smaller  
 1210 values of  $\beta$  to converge to the optimal solution. This is because  
 1211 the system with smaller values of  $\beta$  (i.e.,  $\beta \times T_n$  decreases  
 1212 faster) “jumps” with larger steps at the beginning of the process  
 1213 that may visit several “bad” states before converging to the  
 1214 optimal solution. On the other hand, with greater values of

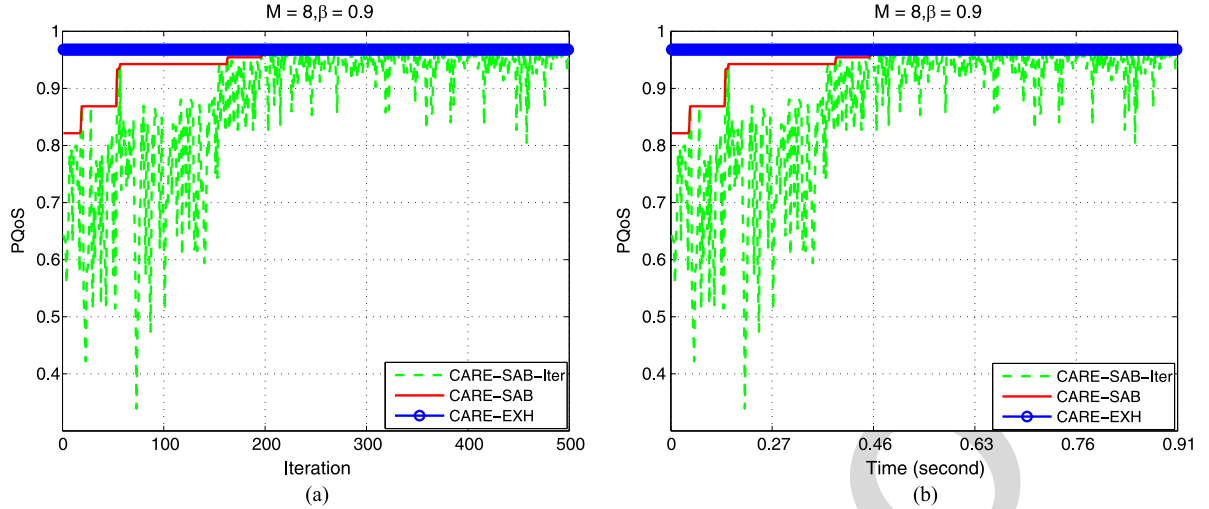


Fig. 10. CARE-SAB convergence versus (a) iteration and (b) time (in seconds):  $M = 8$ ,  $T_0 = 100$ , and  $\beta = 0.9$ .

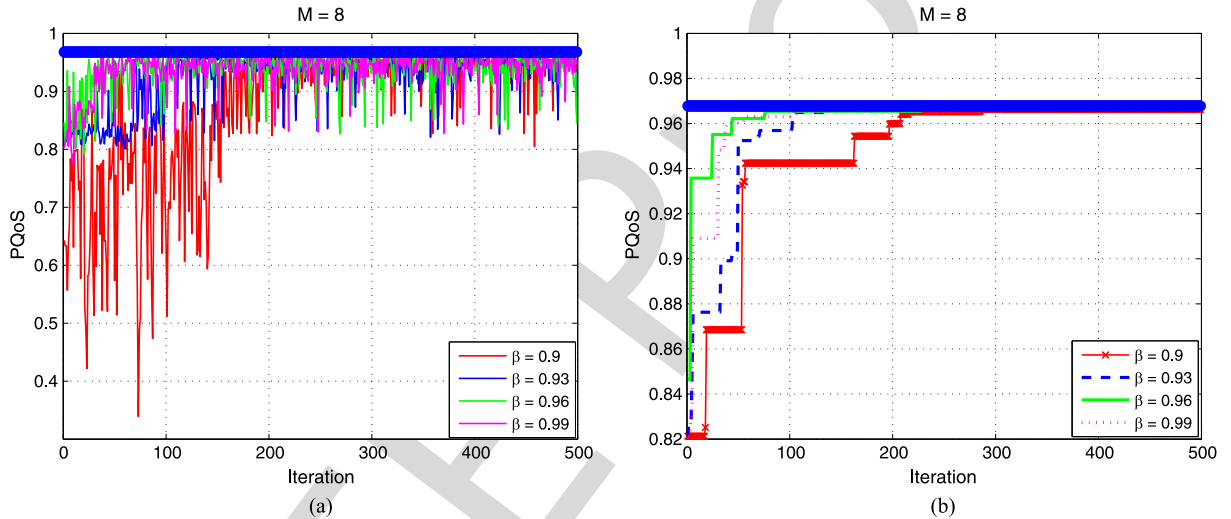


Fig. 11. CARE-SAB convergence for different values of  $\beta$ ,  $M = 8$ ,  $T_0 = 100$ .

1215  $\beta$ , the system is more conservative in giving high probability  
 1216 for exploring “good” states. This setting could reduce the  
 1217 “burning” time of the process before convergence, but it may  
 1218 result in a local optimal solution. Our experiments illustrated  
 1219 in Fig. 11(a) shows that the system quickly converges to the  
 1220 optimal solution within 300 iterations, for all implemented  
 1221 values of  $\beta$ . Fig. 11(b) illustrates the maximal value of PQoS  
 1222 that has been found up to the current iteration. Additionally, the  
 1223 result shows that, for some case, e.g.,  $\beta = 0.99$ , the algorithm  
 1224 quickly obtains the optimal solution in only 100 iterations  
 1225 (about 2.5% of the search space).

1226 Similar performance is obtained for the case of  $M = 10$ ,  
 1227 as illustrated in Fig. 12. As we can observe, the proposed  
 1228 algorithm CARE-SAB quickly converges to the optimal solu-  
 1229 tion and is robust with respect to the values of  $\beta$ . For some  
 1230 cases, e.g.,  $\beta = 0.9$ , the CARE-SAB needs more iterations  
 1231 to find the optimal solution (about 450 iterations). However,  
 1232 we should note that, when  $M = 10$ , there could have 115 975  
 1233 possible partition policies. Therefore, the increase is justifiable  
 1234 compared with the exponential increase of the search space.

1235 Additionally, we further evaluate the convergence rate of  
 1236 the proposed CARE-SAB with different values of the initial  
 1237 temperature  $T_0$ . Fig. 13(a) and (b) illustrates the convergence  
 1238 of CARE-SAB-Iter and CARE-SAB for  $M = 8$  and  $\beta = 0.9$ ,  
 1239 respectively. As illustrated, the proposed algorithm is robust  
 1240 with respect to different initial values of  $T_0$ . We further observe  
 1241 that initializing  $T_0$  with different values only slightly affects  
 1242 the system convergence rate. This is an expected result and  
 1243 consistent with the theoretical analysis because the temperature  
 1244 controls how the algorithm explores the search space. In partic-  
 1245 ular, greater value of  $T_0$  provides more freedom to the algorithm  
 1246 to explore more states, resulting in longer convergence time.  
 1247 However, such a setting will ensure that the global optimal so-  
 1248 lution will be “hit” with high probability. Similar results are also  
 1249 obtained for the case of  $M = 10$  flows, as illustrated in Fig. 14.  
 1250 With larger search space, in the worst case ( $T_0 = 100$ ), the algo-  
 1251 rithm requires about 400 iterations to find the optimal solution.

1252 Finally, we evaluate the scalability and efficiency of the  
 1253 proposed CARE-SAB in Fig. 15. In particular, Fig. 15(a) compares  
 1254 the performance of the CARE-SAB using different

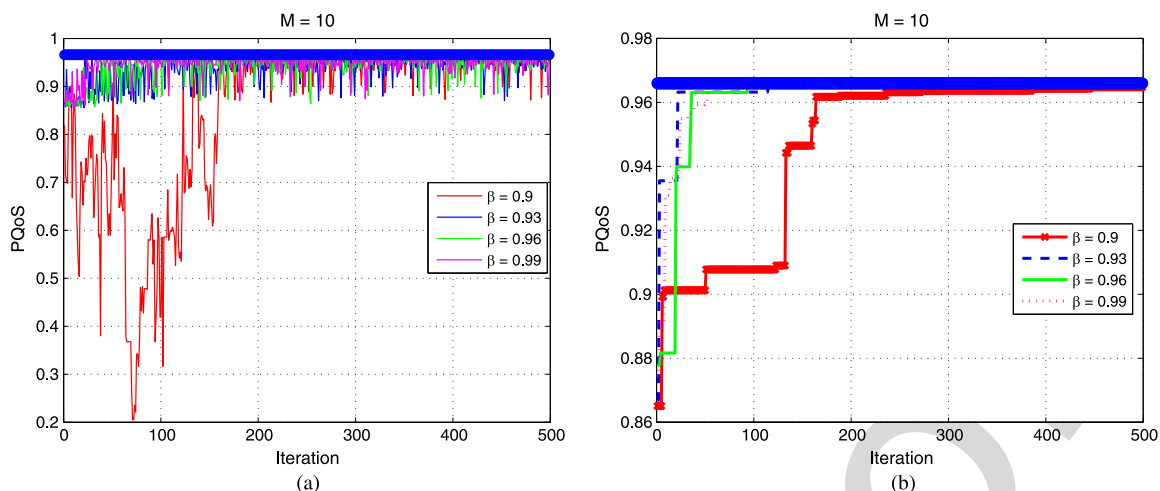


Fig. 12. CARE-SAB convergence for different values of  $\beta$ ,  $M = 10$ ,  $T_0 = 100$ .

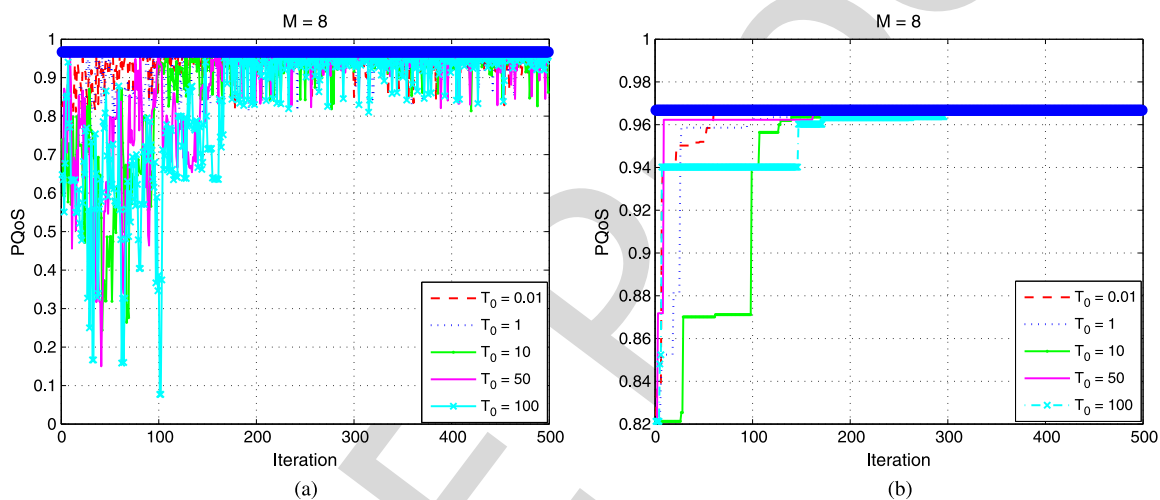


Fig. 13. CARE-SAB convergence for different values of  $T_0$ ,  $M = 8$ ,  $\beta = 0.9$ .

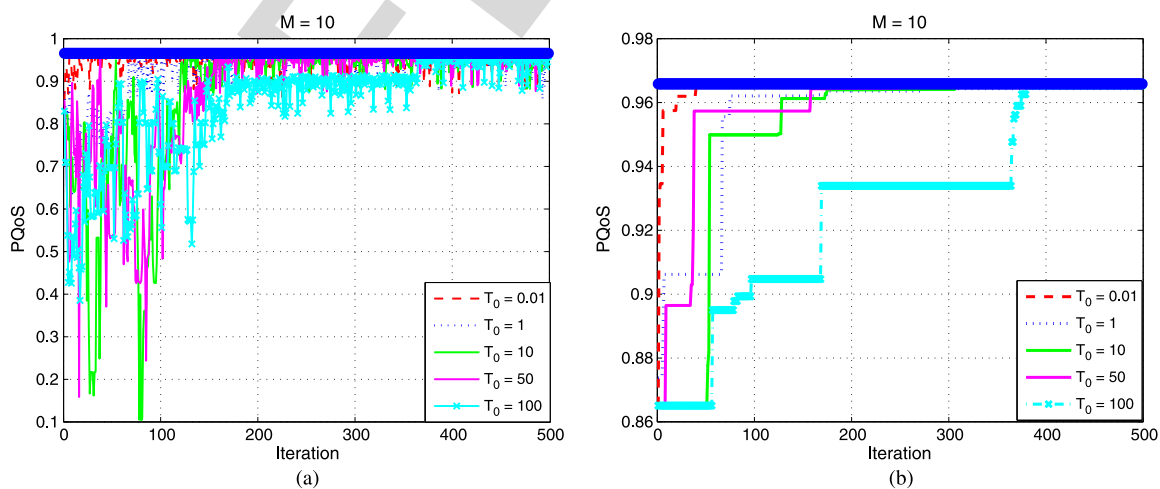


Fig. 14. CARE-SAB convergence for different values of  $T_0$ ,  $M = 10$ ,  $\beta = 0.9$ .

1255 values of iterations with the exhaustive search CARE-EXH  
 1256 versus  $M$ , where CARE-10 and CARE-50 represent CARE-  
 1257 SAB that uses 10 and 50 iterations, respectively. As expected,  
 1258 when  $M$  increases and given a fixed number of iterations,  
 1259 the performance of CARE-SAB schemes decreases due to the

larger search space. However, it is interesting to observe that 1260  
 CARE-SAB schemes can achieve about 90% performance of 1261  
 the exhaustive search despite using a fixed number of iterations. 1262  
 The results illustrate that the designed algorithm is scaled well 1263  
 with the problem size. 1264

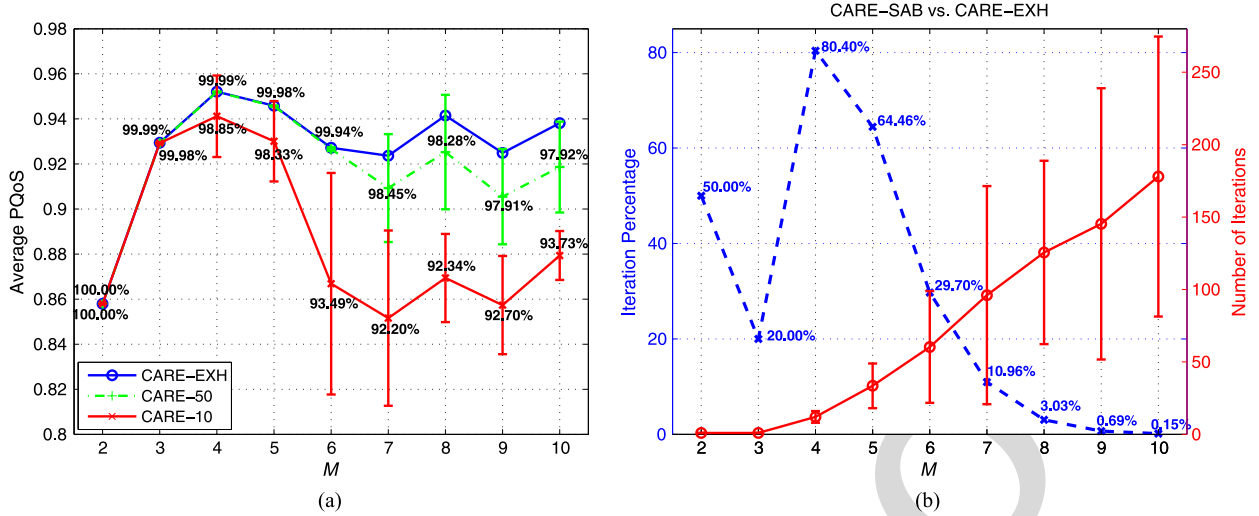


Fig. 15. Evaluating the scalability and robustness of CARE-SAB with  $T_0 = 1$  and  $\beta = 0.9$ : (a) average PQoS for different numbers of iterations and (b) number of iterations for CARE-SAB to obtain optimal partition policy and percentage compared with CARE-EXH.

1265 We further evaluate the scalability of CARE-SAB algorithm  
 1266 by changing  $M$  in Fig. 15(b). In particular, we compute the  
 1267 average number of iterations that the CARE-SAB algorithm  
 1268 consumes to find an optimal solution. In our experiment, for  
 1269 each  $M$ , we run the algorithm many times and compute the  
 1270 average of the results. Fig. 15 illustrates the average number  
 1271 of iterations of CARE-SAB and the corresponding percentage  
 1272 compared with CARE-EXH for  $M = 2, \dots, 10$ . As expected,  
 1273 the number of iterations of CARE-SAB (i.e., the red line)  
 1274 increases with  $M$ , due to the exponential increase of the search  
 1275 space. However, compared with the exhaustive search CARE-  
 1276 EXH, the number of states that CARE-SAB explores to find  
 1277 the optimal solution decreases significantly (i.e., the dash blue  
 1278 curve), i.e., from 50% for  $M = 2$  to 3.03% for  $M = 8$  and to  
 1279 0.15% for  $M = 10$ . The simulation results also confirm that the  
 1280 proposed CARE-SAB algorithm is also scaled well with  $M$ .

## 1281 VII. CONCLUSION

1282 We have investigated the problem of mixing data of traffic  
 1283 with different service classes to improve the network QoS.  
 1284 We first showed that exhaustively mixing data across different  
 1285 data flows at every opportunity may substantially decrease  
 1286 the network QoS. We then proposed CARE, a context-aware  
 1287 interflow network coding and scheduling, to maximize the QoS  
 1288 across all the receivers based on the user satisfaction PQoS. The  
 1289 objective function of CARE is formulated by considering not  
 1290 only the characteristics of traffic but also the service classes and  
 1291 channel conditions. We then showed the hardness of finding an  
 1292 optimal solution to CARE and proposed an efficient approxima-  
 1293 tion algorithm, i.e., CARE-SAB, to obtain a guaranteed near-  
 1294 optimal solution. We further proved the correctness and derived  
 1295 an upper bound on the convergence time of the CARE-SAB  
 1296 algorithm. In addition, we described the PCARE scheme that  
 1297 partially combines data of different flows to further improve  
 1298 the network performance. Simulation results showed that up to  
 1299 a 50% performance gain of the proposed CARE-based schemes  
 1300 can be achieved compared with the existing approaches (e.g.,

RNC). The results also showed that the approximation algo- 1301  
 1302 rithm is robust with respect to the heuristic parameters and  
 1303 well scaled with the number of data flows. To the best of  
 1304 our knowledge, this work is one of a few works studying NC  
 1305 from the QoS point of view. One of our future extensions  
 1306 is to investigate an efficient algorithm for optimal subflow  
 1307 partitioning and encoding of the PCARE scheme.

## 1308 APPENDIX

### 1309 PROOF OF THEOREM 5.2

1310 Let us consider any two states  $\pi_i, \pi_j \in \Omega$ . According to the  
 1311 proposed target distribution, we have  $\theta(\pi_i) = Ce^{S(\pi_i)/T_B}$  and  
 1312  $\theta(\pi_j) = Ce^{S(\pi_j)/T_B}$ . Therefore, we have two possibilities. 1312

1313 Case 1: If  $S(\pi_i) \leq S(\pi_j)$ , we first consider the direction  
 1314 moving from state  $\pi_i$  to state  $\pi_j$ . From (31), we  
 1315 have  $\alpha(\pi_i, \pi_j) = 1$ . Thus 1315

$$\theta(\pi_i)P(\pi_i, \pi_j) = Ce^{\frac{S(\pi_i)}{T_B}} q(\pi_i, \pi_j). \quad (\text{A.1})$$

1316 For the direction from state  $\pi_j$  to state  $\pi_i$ , we have

$$\alpha(\pi_j, \pi_i) = e^{\frac{S(\pi_i) - S(\pi_j)}{T_B}}. \quad (\text{A.2})$$

1317 Hence

$$\begin{aligned} \theta(\pi_j)P(\pi_j, \pi_i) &= Ce^{\frac{S(\pi_j)}{T_B}} q(\pi_j, \pi_i) \alpha(\pi_j, \pi_i) \\ &= Ce^{\frac{S(\pi_j)}{T_B}} q(\pi_j, \pi_i) e^{\frac{S(\pi_i) - S(\pi_j)}{T_B}} \\ &= Ce^{\frac{S(\pi_i)}{T_B}} q(\pi_j, \pi_i). \end{aligned} \quad (\text{A.3})$$

1318 Since  $q(\pi_i, \pi_j) = q(\pi_j, \pi_i)$ , therefore, from (A.1)  
 1319 and (A.3), we obtain the detailed balance 1319  
 equation. 1320

1321 Case 2: Now consider the scenario where  $S(\pi_i) > S(\pi_j)$ .  
 1322 Similarly, from (31), we have 1322

$$\begin{aligned} \alpha(\pi_i, \pi_j) &= e^{\frac{S(\pi_j) - S(\pi_i)}{T_B}} \\ \alpha(\pi_j, \pi_i) &= 1. \end{aligned}$$



1323 Thus

$$\begin{aligned}\theta(\pi_i)P(\pi_i, \pi_j) &= C e^{\frac{S(\pi_i)}{T_B}} q(\pi_i, \pi_j) \\ &= C e^{\frac{S(\pi_i)}{T_B}} q(\pi_j, \pi_i) e^{\frac{S(\pi_j) - S(\pi_i)}{T_B}} \\ &= C e^{\frac{S(\pi_j)}{T_B}} q(\pi_j, \pi_i)\end{aligned}\quad (\text{A.4})$$

$$\begin{aligned}\theta(\pi_j)P(\pi_j, \pi_i) &= C e^{\frac{S(\pi_j)}{T_B}} q(\pi_j, \pi_i) \alpha(\pi_j, \pi_i) \\ &= C e^{\frac{S(\pi_j)}{T_B}} q(\pi_j, \pi_i).\end{aligned}\quad (\text{A.5})$$

1324 From (A.4) and (A.5), we have the detailed bal-  
1325 ance equation; therefore, the theorem follows. ■

## 1326 REFERENCES

- 1327 [1] R. Ahlswede, N. Cai, R. Li, and R. W. Yeung, "Network information  
1328 flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 6, pp. 1204–1216, Jul. 2000.
- 1329 [2] D. Nguyen, T. Tran, T. Nguyen, and B. Bose, "Wireless broadcast with  
1330 network coding," *IEEE Trans. Veh. Technol.*, vol. 58, no. 2, pp. 914–925,  
1331 Feb. 2009.
- 1332 [3] A. Eryilmaz, A. Ozdaglar, and M. Medard, "On delay performance gains  
1333 from network coding," in *Proc. 40th Annu. Conf. Inf. Sci. Syst.*, 2006,  
1334 pp. 864–870.
- 1335 [4] T. Ho, M. Medard, M. Effros, and D. R. Karger, "The benefits of coding  
1336 over routing in a randomized setting," in *Proc. IEEE Symp. Inf. Theory*,  
1337 2003, p. 442.
- 1338 [5] S. Katti *et al.*, "XORs in the air: Practical wireless network coding," in  
1339 *Proc. ACM SIGCOMM*, 2006, pp. 243–254.
- 1340 [6] D. Nguyen, T. Nguyen, and X. Yang, "Wireless multimedia transmis-  
1341 sion with network coding," in *Proc. Packet Video Workshop*, 2007,  
1342 pp. 326–335.
- 1343 [7] Y. Wu, P. A. Chou, and S.-Y. Kung, "Information exchange in wireless  
1344 networks with network coding and physical-layer broadcast," Microsoft  
1345 Res., Tech. Rep. MSR-TR-2004-78, 2004.
- 1346 [8] Y. Wu, P. A. Chou, and S.-Y. Kung, "Minimum-energy multicast in mo-  
1347 bile ad hoc networks using network coding," in *Proc. IEEE Inf. Theory  
1348 Workshop*, 2004, pp. 304–309.
- 1349 [9] J. Widmer, C. Fragouli, and J.-Y. Le Boudec, "Low-complexity energy-  
1350 efficient broadcasting in wireless ad-hoc networks using network coding,"  
1351 in *Proc. Workshop Netw. Coding, Theory, Appl.*, 2005, pp. 1–6.
- 1352 [10] S. R. Chandran and S. Lin, "Selective-repeat-ARQ schemes for broadcast  
1353 links," *IEEE Trans. Commun.*, vol. 40, no. 1, pp. 12–19, Jan. 1992.
- 1354 [11] J. L. Wang and J. A. Silvester, "Performance optimization of the go-back-N  
1355 ARQ protocols over broadcast channels," *Comput. Commun.*, vol. 14,  
1356 no. 7, pp. 393–404, Sep. 1991.
- 1357 [12] T. Tran, T. Nguyen, and B. Bose, "A joint network-channel coding tech-  
1358 nique for single-hop wireless networks," in *Proc. 4th Workshop Netw.  
1359 Coding, Theory, Appl.*, 2008, pp. 1–6.
- 1360 [13] T. Tran, T. Nguyen, B. Bose, and V. Gopal, "A hybrid network coding  
1361 technique for single-hop wireless networks," *IEEE J. Sel. Topics Adv.  
1362 Commun.*, vol. 27, no. 5, pp. 685–698, Jun. 2009.
- 1363 [14] H. Seferoglu and A. Markopoulou, "Opportunistic network coding for  
1364 video streaming over wireless," in *Proc. Packet Video Workshop*, 2007,  
1365 pp. 191–200.
- 1366 [15] H. Seferoglu and A. Markopoulou, "Video-aware opportunistic network  
1367 coding over wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 27,  
1368 no. 5, pp. 713–728, Jun. 2009.
- 1369 [16] J. Domzal, J. Dudek, P. Jurkiewicz, L. Romanski, and R. Wojcik, "The  
1370 cross-protect router: Implementation tests and opportunities," *IEEE Com-  
1371 mun. Mag.*, vol. 52, no. 9, pp. 115–123, Sep. 2014.
- 1372 [17] J. W. Roberts, "Internet traffic, QoS, and pricing," *Proc. IEEE*, vol. 92,  
1373 no. 9, pp. 1389–1399, Sep. 2004.
- 1374 [18] G. Miao and Z. Niu, "Bandwidth management for mixed unicast and  
1375 multicast multimedia flows with perception based QoS differentiation,"  
1376 in *Proc. IEEE ICC*, 2006, pp. 687–692.
- 1377 [19] T. Tran and T. Nguyen, "Adaptive network coding for wireless access  
1378 networks," in *Proc. 19th ICCCN*, Aug. 2010, pp. 1–6.
- 1379 [20] G. Perboli, R. Tadei, and L. Gobatto, "The multi-handler knapsack prob-  
1380 lem under uncertainty," Interuniversity Res. Cent., Montreal, QC, Canada,  
1381 Rep. CIRRELT-2012-69, 2012.
- 1382 [21] A. Sinclair, "Improved bounds for mixing rates of Markov chains and  
1383 multicommodity flow," in *Combinatorics, Probability and Computing*.  
1384 Berlin, Germany: Springer-Verlag, 1992.
- [22] R. W. Yeung, S. Y. R. Li, and N. Cai, "Linear network coding," *IEEE* 1385  
*Trans. Inf. Theory*, vol. 49, no. 2, pp. 371–381, Feb. 2003. 1386
- [23] R. Koetter and M. Medard, "An algebraic approach to network coding," 1387  
*IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 782–795, Oct. 2003. 1388
- [24] M. Medard, M. Effros, T. Ho, and D. Karger, "On coding for non- 1389  
multicast networks," in *Proc. Allerton Conf. Commun.*, 2003, pp. 1–7. 1390
- [25] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Proc. 41st* 1391  
*Allerton Conf. Commun., Control Comput.*, Monticello, IL, USA, 2003, 1392  
pp. 1–10. 1393
- [26] T. Ho *et al.*, "A random linear network coding approach to multicast," 1394  
*IEEE Trans. Inf. Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006. 1395
- [27] S. Katti, D. Katabi, W. Hu, H. Rahul, and M. Medard, "The importance of 1396  
being opportunistic: Practical network coding for wireless environments," 1397  
in *Proc. 43rd Annu. Allerton Conf. Commun.*, 2005, pp. 1–10. 1398
- [28] C. Fragouli, J. Le Boudec, and J. Widmer, "Network coding: An instan- 1399  
t primer," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 1, 1400  
pp. 63–68, Jan. 2006. 1401
- [29] S. Deb *et al.*, "Network coding for wireless applications: A brief tutorial," 1402  
in *Proc. IWWAN*, 2005, pp. 196–200. 1403
- [30] A. Douik, S. Sorour, M. Alouini, and T. Al-Naffouri, "Delay minimization 1404  
for instant decodable network coding in persistent channels with feedback 1405  
intermittence," *ArXiv e-Prints*, 2013. 1406
- [31] A. Douik, S. Sorour, M. Alouini, and T. Al-Naffouri, "Delay reduction in 1407  
lossy intermittent feedback for generalized instantly decodable network 1408  
coding," in *Proc. IEEE Int. Conf. WiMob Comput., Netw. Commun.*, 2013, 1409  
pp. 388–393. 1410
- [32] R. Koetter and M. Medard, "Beyond routing: An algebraic approach to 1411  
network coding," in *Proc. IEEE INFOCOM*, 2002, pp. 122–130. 1412
- [33] Default TTL Values in TCP/IP, 2013. [Online]. Available: [http://www.](http://www.map.meteoswiss.ch/map-doc/ftp-probleme.htm) 1413  
[map.meteoswiss.ch/map-doc/ftp-probleme.htm](http://www.map.meteoswiss.ch/map-doc/ftp-probleme.htm) 1414
- [34] IP Option Numbers [RFC791], [RFC1122], 2013. [Online]. Available: 1415  
<http://www.iana.org/assignments/ip-parameters/ip-parameters.xml> 1416
- [35] I. Hou and P. R. Kumar, "Scheduling heterogeneous real-time traffic over 1417  
fading wireless channels," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9. 1418
- [36] M. Mazzotti *et al.*, "Analysis of packet-level forward error correction for 1419  
video transmission," in *Proc. IEEE VTC Spring*, 2011, pp. 1–5. 1420
- [37] C. Feng and B. Li, "Network coding for content distribution and multime- 1421  
dia streaming in P2P networks," in *Network Coding: Fundamentals and* 1422  
*Applications*. San Diego, CA, USA: Academic, 2012. 1423
- [38] S. Sorour, A. Douik, S. Valaee, T. Y. Al-Naffouri, and M.-S. Alouini, "Parti- 1424  
ally blind instantly decodable network codes for lossy feedback environ- 1425  
ment," *IEEE Trans. Wireless Commun.*, vol. 13, no. 9, pp. 4871–4883, 1426  
Sep. 2014. 1427
- [39] Y. Wu, J. Padhye, R. Chandra, V. Padmanabhan, and P. A. Chou, "The 1428  
local mixing problem," in *Proc. IEEE Inf. Theory Appl. Workshop*, 2006, 1429  
pp. 1–5. 1430
- [40] A. Shokrollahi, "Raptor codes," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, 1431  
pp. 2551–2567, Jun. 2006. 1432
- [41] T. Ho, M. Medard, J. Shi, M. Effros, and D. R. Karger, "On randomized 1433  
network coding," in *Proc. 41st Annu. Allerton Conf. Commun., Control,* 1434  
*Comput.*, 2003, pp. 1–10. 1435
- [42] R. Aravind, M. Civanlar, and A. Reibman, "Packet loss resilience of 1436  
MPEG-2 scalable video coding algorithms," *IEEE Trans. Circuits Syst.* 1437  
*Video Technol.*, vol. 6, no. 5, pp. 426–435, Oct. 1996. 1438
- [43] R. Singh, A. Ortega, L. Perret, and W. Jiang, "Comparison of multiple 1439  
description coding and layered coding based on network simulations," in 1440  
*Proc. Dispersivity Routing Store Forward Netw.*, 2000, pp. 929–939. 1441
- [44] A. Silberschatz, P. Galvin, and G. Gagne, *Operating System Concepts: 1442  
Process Scheduling*. Hoboken, NJ, USA: Wiley, 2010. 1443
- [45] M. Ghaderi, D. Towsley, and J. Kurose, "Reliability benefit of network 1444  
coding," *Comput. Sci. Dept., Univ. Mass. Amherst*, Amherst, MA, USA, 1445  
Tech. Rep. 07-08, 2007. 1446
- [46] M. Jerrum and A. Sinclair, "The Markov chain Monte Carlo method: An 1447  
approach to approximate counting and integration," in *Approximation* 1448  
*Algorithms for NP-hard Problems*, D. S. Hochbaum ed. Boston, MA, 1449  
USA: PWS, 1996. 1450
- [47] M. Abramowitz and I. A. Stegun, Eds., "Handbook of Mathematical 1451  
Functions With Formulas, Graphs, and Mathematical Tables," in *The 9th* 1452  
*Printing*. New York, NY, USA: Dover, 1972. 1453
- [48] Voice Over IP—Per Call Bandwidth Consumption. [Online]. Available: 1454  
[http://www.cisco.com/c/en/us/support/docs/voice/voice-quality/7934-](http://www.cisco.com/c/en/us/support/docs/voice/voice-quality/7934-bwidth-consume.html) 1455  
[bwidth-consume.html](http://www.cisco.com/c/en/us/support/docs/voice/voice-quality/7934-bwidth-consume.html) 1456
- [49] C. Wang, D. Koutsonikolas, Y. Hu, and N. Shroff, "FEC-based AP down- 1457  
link transmission schemes for multiple flows: Combining the reliability 1458  
and throughput enhancement of intra- and inter-flow coding," *Perform.* 1459  
*Eval.*, vol. 68, no. 11, pp. 1118–1135, Nov. 2011. 1460

AQ11 1461  
1462  
1463  
1464  
1465  
1466  
1467  
1468  
1469  
1470  
1471



**Tuan Tran** (M'XX) received the B.S. degree from Hanoi University of Science and Technology, Hanoi, Vietnam, in 2000; the M.S. degree from the Polytechnic University of Turin, Turin, Italy, in 2006; and the Ph.D. degree from Oregon State University, Corvallis, OR, USA, in 2010, all in computer engineering.

He is currently an Assistant Professor with the College of Computer and Information Technology, Sullivan University, Louisville, KY, USA. Prior to that, he was a Postdoctoral Scholar with Arizona

State University, Tempe, AZ, USA, and the University of Louisville from 2010 to 2012. His research interests include computer networks and cybersecurity, stochastic system modeling, network and channel coding, wireless systems, and multimedia networking.

Dr. Tran received the Best Paper Runner-Up Award at the IEEE International Conference on Computer Communication Networks in 2010 and the Jack Neubauer Memorial Award for the Best Systems Paper published in the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY in 2012.



**Thinh Nguyen** (M'04) received the B.S. degree from 1480 the University of Washington, Seattle, WA, USA, in 1481 1995 and the Ph.D. degree from the University of 1482 AQ12 California, Berkeley, CA, USA, in 2003. 1483

He is currently an Associate Professor with the 1484 School of Electrical Engineering and Computer Sci- 1485 ence, Oregon State University, Corvallis, OR, USA. 1486 He is interested in all things stochastic, with ap- 1487 plications to signal processing, distributed systems, 1488 wireless networks, network coding, and quantum 1489 walks. 1490

Dr. Nguyen has served as an Associate Editor for the IEEE TRANSACTIONS 1491 ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY and the IEEE 1492 TRANSACTIONS ON MULTIMEDIA. 1493

IEEE PROOF