Network Codes for Real-Time Applications

Anh Le, *Member, IEEE,* Arash S. Tehrani, *Member, IEEE,*
Alexandros G. Dimakis, *Member, IEEE,* and Athina Markopoulou, *Senior Member, IEEE*

Abstract

We consider the scenario of broadcasting for real-time applications and loss recovery via instantly decodable network coding. Past work focused on minimizing the completion delay, which is not the right objective for real-time applications that have strict deadlines. In this work, we are interested in finding a code that is instantly decodable by the maximum number of users. First, we prove that this problem is NP-Hard in the general case. Then we consider the practical probabilistic scenario, where users have i.i.d. loss probability and the number of packets is linear or polynomial in the number of users. In this scenario, we provide a polynomial-time (in the number of users) algorithm that finds the optimal coded packet. The proposed algorithm is evaluated using both simulation and real network traces of a real-time Android application. Both results show that the proposed coding scheme significantly outperforms the state-of-the-art baselines: an optimal repetition code and a COPE-like greedy scheme.

Index Terms

Broadcast, Loss Recovery, Instantly Decodable Codes, Real-Time Applications, Network Coding.

I. Introduction

Broadcasting data to multiple users is widely used in several wireless applications, ranging from satellite communications to WiFi networks. Wireless transmissions are subject to packet

A. Le and A. Markopoulou are with the Department of Electrical Engineering and Computer Science, University of California, Irvine, CA, 92697.

E-mail: {anh.le, athina}@uci.edu

A. S. Tehrani is with the Department of Electrical Engineering, University of Southern California, CA, 90089.

E-mail: arash.sabertehrani@usc.edu

A. G. Dimakis is with the Department of Electrical and Computer Engineering, University of Texas, Austin, TX, 78712.

losses due to channel impairments, such as, wireless fading and interference. Previous work has shown that coding can improve transmission efficiency, throughput, and delay over broadcast erasure channels [1]–[6]. Intuitively, the diversity of lost packets across different users creates coding opportunities that can improve various performance metrics.

In this work, we are interested in packet recovery for real-time applications, such as, fast-paced multi-player games and live video streaming. Real-time applications have two distinct characteristics: (i) they have strict and urgent deadlines, *i.e.*, a packet is outdated after a short amount of time, and (ii) they can tolerate some losses, *e.g.*, a game client can restore its state in the presence of losses by resyncing periodically [7]. Although having limited fault tolerance, these applications may suffer significantly from packet losses and lead to poor performance, *e.g.*, jittery game animation and low quality video playback. Hence, it is highly desirable to recover packet losses with very low delay and within a very narrow coding window. Motivated by the above observations, we focus on coding schemes for loss recovery that allows instantaneous decoding, *i.e.*, with zero delay. These coding schemes are also known as Instantly Decodable Network Codes (IDNC).

Previous work on IDNC [2]–[4], [8]–[10] focused on minimizing the completion delay, *i.e.*, the time it takes to recover all the losses at all users. We formulate a different problem that is more relevant to real-time applications, called *Real-Time IDNC*: Consider a source that broadcasts a set of packets, $\mathcal{X}$, to a set of users, $\mathcal{U}$. Each user, $u \in \mathcal{U}$, wants all packets in $\mathcal{X}$ and already knows a subset of them, $\mathcal{H}_u \subset \mathcal{X}$, for example, through previous transmissions. The goal is to choose one (potentially coded) packet to broadcast from the source, so as to maximize the number of users who can immediately recover one lost packet. This problem is highly relevant in practice, yet – to the best of our knowledge – only solved in heuristic ways so far, *e.g.*, see [2], [11]. Our main contributions are the following:

- We show that Real-Time IDNC is NP-hard. To do so, we first map Real-Time IDNC to the Maximum Clique problem in an IDNC graph (to be precisely defined in Section III). We then show that the Maximum Clique problem is equivalent to an Integer Quadratic Program (IQP) formulation. Finally, we provide a reduction from a well-known NP-Hard problem (the Exact Cover by 3-Sets) to this IQP problem.
- We analyze random instances of the problem, where each packet is successfully received by

each user randomly and independently with the same probability. This problem, referred to as Random Real-Time IDNC, corresponds to a Maximum Clique problem on an appropriately created random IDNC graph. Surprisingly, we show that when the number of packets is linear or polynomial in the number of users, the Maximum Clique problem can be solved with high probability on this particular family of random graphs, by a polynomial-time (in the number of users) algorithm, which we refer to as the *Max Clique* algorithm.

We implement and compare the proposed coding scheme, Max Clique, against two baselines: an optimal repetition code and a COPE-like greedy scheme proposed in [2]. Simulations show that Max Clique significantly outperforms these state-of-the-art schemes over a range of scenarios, for the loss probability varying from .01 to .99. For example, for 20 users and 20 packets, Max Clique improves by a factor of 1.3 on average, and performs up to 1.6 times better than the COPE-like code and up to 3.8 times better than the optimal repetition code. Finally, we evaluate Max Clique on network traces of a real-time multi-player game on Android that uses broadcast. The results of this trace-based evaluation confirm the superior performance of Max Clique over the baselines.

The remainder of this paper is organized as follows. Section II discusses related work. Section III formulates the problem. Section IV describes the maximum clique and integer program formulations as well as the proof of NP-completeness. Section V analyzes the probabilistic version (Random Real-Time IDNC problem) and describes Max Clique, the polynomial-time algorithm to find a maximum clique w.h.p. Section VI evaluates and compares our coding scheme with existing schemes. Section VII concludes the paper.

II. Related Work

Instantly Decodable Network Coding. Katti *et al.* [11] proposed COPE, an opportunistic inter-session network coding scheme for wireless networks. Encoded packets are chosen so that they are immediately decodable at the next hop. The algorithm considers combining packets in a FIFO way (first-in-first-out, as stored in the transmitting queue) and greedily maximizes the number of receivers that can decode in the next time slot. Keller *et al.* [2] investigated algorithms that minimize decoding delay, including two algorithms that allow for instantaneous decoding: a COPE-like greedy algorithm and a simple repetition algorithm. In Section VI, we use these two algorithms as baselines for comparison.

In [3], Sadeghi *et al.* improved the opportunistic algorithm previously proposed in [2] by giving high priority to packets that are needed by a large number of users. The authors also gave an Integer Linear Program formulation to the problem of finding an instantly decodable packet that maximizes the number of beneficiary users. Furthermore, they showed that it is NP-hard based on the *Set Packing* problem. We note that their formulation differs from ours since it requires that a coded packet must be instantly decodable by *all* users, where some users may not benefit from the packet. This may lead to a suboptimal solution because there may be a coded packet that is only instantly decodable *by some but not all* users but is beneficial to a larger number of users. Our formulation ensures that we find this optimal packet.

Sorour *et al.* have an extensive line of work investigating instantly decodable codes [4], [8]–[10], [12]–[14], focusing on minimizing the completion delay. They introduced the term Instantly Decodable Network Coding (IDNC) that we adopt in this work. In [12], they proposed a construction of IDNC graphs based on feedback from the users and then introduced a transmission scheme based on graph partitioning. We consider the same construction of IDNC graphs as in [12]. Based on a stochastic shortest path formulation, they proposed a heuristic algorithm to minimize the completion delay [4]. In [8], they introduced the notion of *generalized* IDNC problem, which does not require the transmitted code to be decodable by all users, as opposed to the *strict* version studied previously [2], [4], [12]. Real-Time IDNC considers the generalized version. Furthermore, in [8], they related finding an optimal IDNC code to the Maximum Clique problem in IDNC graphs and suggested that it is NP-Hard; however, no explicit reduction was provided. In [9] and [10], they extended [4] to cope with limited or lossy feedback. In [13], they considered the case of multicast instead of broadcast, and in [14], the case where users could buffer coded packets in addition to plain packets was investigated.

Li *et al.* [15] adopted IDNC for video streaming and showed that, for independent channels and sufficiently large video file, their proposed IDNC schemes are asymptotically throughput-optimal subject to hard deadline constraints when there are *no more than three users*. In contrast, we consider an arbitrary number of users, and we provide the optimal single transmission.

Index Coding. Our problem setup is relatively similar to that of the Index Coding (IC) problem, introduced by Birk and Kol [16] previously and extensively studied since. An IC problem also considers a base station that knows a set of packets, $\mathcal{X}$, and a set of users. Each user $(x, \mathcal{H})$

demands one particular packet, $x \in \mathcal{X}$, and has side information consisting of a subset of packets, $\mathcal{H} \subset \mathcal{X}$. The base station broadcasts to all users without errors. The goal is to find an encoding scheme that minimizes the number of transmissions required to deliver the packets to all users.

It has been shown that except for the cases that can be solved with one or two transmissions, other instances of the IC problem are NP-hard to solve [17]–[19], including a variation of IC where users are pliable and happy to receive any one packet [20], [21]. Furthermore, even finding an approximation to the problem has been shown to be hard [22]. [23], [24] provided heuristic algorithms to find such codes.

Despite the similarities, there are two main differences between our problem and IC. First, in our problem, each user wants *all* the packets, not just a single packet. Second, we want to find an instantly decodable packet that maximizes the number of beneficiary users, not the total number of transmissions to satisfy all users.

Data Exchange. The Data Exchange (DX) problem, originally introduced by El Rouayheb *et al.* [25], also has a similar setup to our problem: There is a set of packets, $\mathcal{X}$, and a set of users $\mathcal{U}$. Each user, $u \in \mathcal{U}$, knows a subset of packets, $\mathcal{H}_u \subset \mathcal{X}$, and wants all packets in $\mathcal{X}$. In DX, there is no base station, and the users broadcast messages. The objective is to find an encoding scheme that minimizes the number of transmissions required to deliver all packets in $\mathcal{X}$ to all users.

To solve the DX problem, a randomized polynomial-time solution was proposed in [26], and deterministic polynominal-time solutions were proposed in [29] and [27]. [31] studied the problem in general network topologies. Variants of the problem where there are helpers and transmission weights were studied in [28] and [30]. Various necessary and sufficient conditions that characterize feasible transmission schemes for the problem were proposed in [32]–[34], under a different name of *universal recovery*.

Similar to DX, in our setting, all users want all the packets in $\mathcal{X}$. However, there are two main differences: (i) in our setting, only the base station can broadcast as opposed to having all users capable of broadcasting, and (ii) we are interested in instantaneous decoding to maximize the number of beneficiary users with one transmission, as opposed to minimizing the total number of transmissions.

This Work in Perspective. A preliminary version of this work has appeared in [35]. In this

paper, we extend the previous work in the following ways: First, we provide complete proofs of all theorems and propositions. Second, we collect network traces of a real-time application that utilizes broadcast and present a new evaluation based on the traces. Finally, we discuss and highlight the similarities and differences between the Real-Time IDNC, Index Coding, and Data Exchange problems.

III. Problem Formulation

Let $\mathcal{U} = \{u_1, \cdots, u_n\}$ denote the set of n users, and $\mathcal{P} = \{p_1, \cdots, p_m\}$ be the set of m packets. We assume that the original m packets were broadcast by a base station. Due to packet loss, each of n users missed some of the m packets. We denote the set of packets that were successfully received by user i by $\mathcal{H}_i$. Furthermore, let $\mathcal{W}_i$ be the set of packets that user i still wants, *i.e.*, $\mathcal{W}_i = \mathcal{P} \setminus \mathcal{H}_i$. Consistently with [12], [17], we call $\mathcal{H}$'s and $\mathcal{W}$'s the "Has" and "Want" sets.

After the initial broadcast, the base station tries to recover the losses, $\mathcal{W}$'s, by sending coded packets and exploiting the side information of the already delivered packets, $\mathcal{H}$'s. Let the $n \times m$ matrix $\mathbf{A}$ be the identification matrix for the side information of the users, *i.e.*, entry $a_{ij} = 1$ if user u_i wants packet p_j and 0 otherwise. $\mathbf{A}$ is also called a feedback matrix, as in [4], [8]–[10], [12], [13]. Let us clarify this by an example.

Example 1. Consider a scenario with 3 users and 6 packets. Furthermore, assume that after the initial broadcast, user u_1 successfully received packets p_1 and p_2; user u_2 received p_3 and p_5; and user u_3 received p_3 and p_6. The scenario is depicted in Fig. 1. In this case, the side information matrix is as follows:

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}.$$

To deliver the packets in the Want sets of the users, we focus on instantly decodable, lightweight coding schemes that operate in GF(2). For a set of packets, $\mathcal{M}$, the corresponding coded packet c is their binary sum, denoted by $\bigoplus$:

$$c = \bigoplus_{p_i \in \mathcal{M}} p_i \,. \tag{1}$$

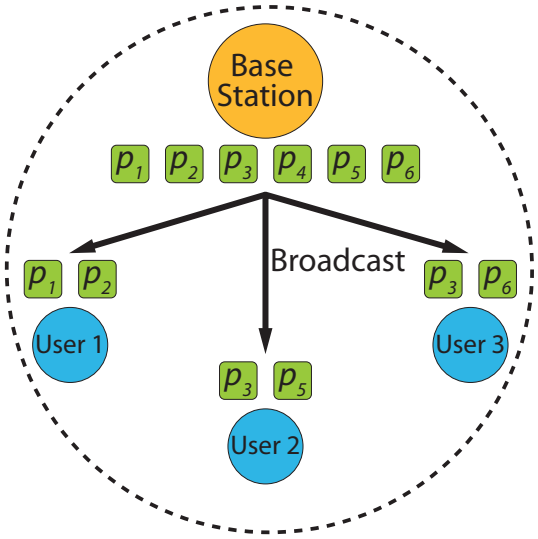


Fig. 1. Example 1: A base station broadcast 6 packets $\{p_1, \cdots, p_6\}$ to 3 users. Due to packet loss, user 1 only received p_1 and p_2; user 2 received p_3 and p_5; user 3 received p_3 and p_6.

Definition 1. *A coded packet, $c^{\mathcal{N}}$, is* instantly decodable *with respect to a set of users, $\mathcal{N}$, if and only if*

(i) Every user, $u_i \in \mathcal{N}$, can decode $c^{\mathcal{N}}$ immediately upon reception to recover a packet $p^i \in \mathcal{W}_i$. That is, each user in $\mathcal{N}$ benefits *from $c^{\mathcal{N}}$ by recovering one of the packets from its want set.*

(ii) Every packet in the binary sum of $c^{\mathcal{N}}$ is wanted by at least one user in $\mathcal{N}$.

For example, for the scenario of Example 1, the coded packet $c^{\{u_1,u_2,u_3\}} = p_1 \oplus p_3$ is instantly decodable with respect to $\{u_1, u_2, u_3\}$ since u_1 can recover p_3, while u_2 and u_3 can get p_1. Meanwhile, $c^{\{u_2,u_3\}} = p_5 \oplus p_6$ is not instantly decodable with respect to u_1. Furthermore, we do not consider $c^{\{u_2,u_3\}} = p_1 \oplus p_5 \oplus p_6$ instantly decodable with respect to $\{u_2, u_3\}$ since although c can be decoded by u_2 and u_3, packet p_1, which is a component of c_3, is not needed by either u_2 or u_3. (From here on, we will omit the superscript $\mathcal{N}$ of $c^{\mathcal{N}}$ when there is no ambiguity.)

We would like the coded packet to be immediately beneficial to as many users as possible. Thus, our notion of optimality is with respect to the cardinality of the set of beneficiary users $|\mathcal{N}|$.

The Real-Time IDNC Problem: Given a side information matrix $\mathbf{A}$, find the optimal instantly

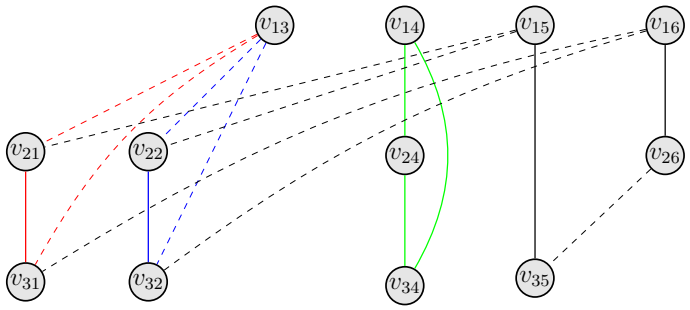


Fig. 2. The Instantly Decodable Network Coding (IDNC) graph of Example 1. Solid edges are edges of type (i) and dashed edges are edges of type (ii). There are three maximum cliques: $\{v_{13}, v_{21}, v_{31}\}$, $\{v_{13}, v_{22}, v_{32}\}$, and $\{v_{14}, v_{24}, v_{34}\}$, all of which are of size 3.

decodable packet $c^{\mathcal{N}}$.

IV. Maximum Cliques in IDNC Graphs

Given a side information matrix $\mathbf{A}$, we form an Instantly Decodable Network Coding (IDNC) graph corresponding to $\mathbf{A}$ as in [12]: We create a vertex v_{ij} when user u_i still wants packet p_j. For instance, for matrix $\mathbf{A}$ in Example 1, there is a vertex for each entry 1 in the matrix. Given a vertex v_{ij}, we use the term *user index* of v_{ij} to indicate i and *packet index* of v_{ij} to indicate j. There is an edge between two vertices v_{ij} and $v_{k\ell}$ if one of the below conditions hold:

(i) $j = \ell$: In this case, both users u_i and u_k wants the same packet $p = p_j = p_\ell$.

(ii) $p_j \in \mathcal{H}_k$ and $p_\ell \in \mathcal{H}_i$: In this case, user u_k has packet p_j that user u_i still wants, and vice versa.

Denote the IDNC graph corresponding to a matrix $\mathbf{A}$ by $G^{\mathbf{A}} = (\mathcal{V}, \mathcal{E})$. Figure 2 shows the IDNC graph corresponding to the side information matrix given in Example 1.

A. Cliques and Instantly Decodable Packets

Proposition 1. *Finding an optimal instantly decodable code given a side information matrix* $\mathbf{A}$ *is equivalent to finding a maximum clique in the corresponding IDNC graph* $G^{\mathbf{A}}$.

We prove this proposition by establishing the following Lemmas 2 and 3. The first lemma states the relationship between instantly decodable packets and cliques in $G^{\mathbf{A}}$.

Lemma 2. *Given a side information matrix* $\mathbf{A}$ *and its IDNC graph* G^A*, an instantly decodable packet has a one-to-one correspondence to a clique in* G^A*.*

The second lemma expresses the relationship between the number of users benefiting from an instantly decodable packet and the size of the clique corresponding to the packet.

Lemma 3. *Given a side information matrix* $\mathbf{A}$ *and its IDNC graph* $G^{\mathbf{A}}$*, let* $c^{\mathcal{N}}$ *be an instantly decodable packet, and let* $\mathcal{C}$ *be the corresponding clique of* $c^{\mathcal{N}}$ *in* G^A*, then* $|\mathcal{C}| = |\mathcal{N}|$*.*

The proofs of Lemma 2 and 3 are provided in Appendices A and B, correspondingly. Intuitively, let us consider the clique involving v_{13}, v_{21}, and v_{31} in Example 1. XORing all packets corresponding to vertices of this clique, *i.e.*, $p_1 \oplus p_3$, forms an instantly decodable packet because (i) user 1 must have p_1, and users 2 and 3 must have p_3, otherwise there are no edges (v_{13}, v_{21}) and (v_{13}, v_{31}), and (ii) each component of the coded packet is wanted by the user corresponding to the row of the vertex. Finally, the clique size equals 3, which is the number of beneficiary users.

B. NP-Completeness

Finding a maximum clique in a general graph is well known to be NP-Hard. This result, however, is not directly applicable to IDNC graphs as they have special structural properties. In this section, we will show that the problem of finding a maximum clique in an IDNC graph is indeed NP-Hard. We show this by first showing that finding a maximum clique in an IDNC graph is equivalent to finding an optimal solution to an Integer Quadratic Programming (IQP) problem. We then describe a reduction from a well known NP-Complete problem, the Exact Cover by 3-Sets problem, to the decision version of the IQP problem.

1) Integer Quadratic Programming Formulation: Given a side information matrix $\mathbf{A}$ of size $n \times m$, we formulate the IQP problem as follows. Let $\mathbf{r}$ be a binary $n \times 1$ vector: $r_i \in \{0, 1\}, i = 1, \cdots, n$. Similarly, let $\mathbf{c}$ be a binary $m \times 1$ vector: $c_j \in \{0, 1\}, j = 1, \cdots, m$. Below is the IQP problem for $\mathbf{A}$:

$$\text{Maximize:} \quad V = \mathbf{r}^T \mathbf{A}\, \mathbf{c} = \sum_{i=1}^{n} \sum_{j=1}^{m} r_i\, c_j\, a_{ij}.$$

$$\text{Subject to:} \quad r_i \sum_{j=1}^{m} c_j\, a_{ij} \leq 1,\ \forall i = 1, \cdots, n\,. \tag{1}$$

$$r_i, c_j \in \{0, 1\}\,. \tag{2}$$

Proposition 4. *Given a side information matrix* $\mathbf{A}$ *and its IDNC graph* G^A*, finding a maximum clique in* $G^{\mathbf{A}}$ *is equivalent to finding an optimal solution to the corresponding IQP.*

We prove this proposition by establishing the following Lemmas 5 and 6. The first lemma expresses the relationship between the above IQP problem and the problem of finding maximum clique in $G^{\mathbf{A}}$.

Lemma 5. *Given a side information matrix* $\mathbf{A}$ *and its IDNC graph* $G^{\mathbf{A}}$*, a clique in* $G^{\mathbf{A}}$ *has a one-to-one correspondence to a feasible solution of the IQP problem for* $\mathbf{A}$.

Proof:

($\Rightarrow$) We first show that a clique in the IDNC graph maps to a feasible pair of vectors $\mathbf{r}$ and $\mathbf{c}$ of the IQP problem, which is uniquely identified by the user and packet indices of the vertices in the clique.

Let $\mathcal{C}$ be a clique in $G^{\mathbf{A}}$: $\mathcal{C} = \{v_{i_1,j_1}, \cdots, v_{i_k,j_k}\}$. Let $\mathcal{I}$ be the set of user indices: $\mathcal{I} = \{i_1, \cdots, i_k\}$, and $\mathcal{J}$ be the set of packet indices: $\mathcal{J} = \{j_1, \cdots, j_k\}$. We create the feasible pair of $\mathbf{r}$ and $\mathbf{c}$ as follows: Set $r_i = 1$ if $i \in \mathcal{I}$ and 0 otherwise, and set $c_j = 1$ if $j \in \mathcal{J}$ and 0 otherwise.

To show that this pair of vectors is a feasible solution, we proceed by showing that condition (1) of the IQP holds for all user indices. Let i be any user index, $i \in [1, n]$. It is clear that (1) holds if $r_i = 0$. When $r_i = 1$, it suffices to show that no two vertices of $\mathcal{C}$ have the same user index. Indeed, this follows from the observation that there is no edge between any two vertices having the same user index (on the same row) in the IDNC graph.

($\Leftarrow$) Next, we show that a feasible solution of the IQP maps to a uniquely identified clique in the IDNC graph. Let the pair of vectors $\mathbf{r}$ and $\mathbf{c}$ be a feasible solution. We map this pair to a clique $\mathcal{C}$ in $G^{\mathbf{A}}$ as follows: Initialize $\mathcal{C} = \emptyset$. For $i \in [1, n]$, for $j \in [1, m]$, if $r_i = c_j = a_{ij} = 1$, add vertex v_{ij} to $\mathcal{C}$.

Now pick any pair of vertices v_{st} and v_{pq} in $\mathcal{C}$. It is clear that if $t = q$, there is an edge between these two vertices. It remains to show that when $t \neq q$, user u_s has packet p_q and user u_p has packet p_t. We will show that user u_s must have packet p_q. The other condition follows by symmetry. Assume otherwise, *i.e.*, user u_s does not have packet p_q, which means $a_{sq} = 1$. Since $\mathcal{C}$ contains v_{st} and v_{pq}, $r_s = c_t = a_{st} = 1$ and $r_p = c_q = a_{pq} = 1$. But then for row s, condition (1) of the IQP problem fails since

$$r_s \sum_{j=1}^{m} c_j \, a_{sj} \geq r_s \, c_t \, a_{st} + r_s \, c_q \, a_{sq} = 2 \, .$$

Finally, it is easy to check that for the above two mappings, one is the reverse of the other. ■

Lemma 6. *Given a side information matrix* $\mathbf{A}$ *and its IDNC graph* $G^{\mathbf{A}}$*, the size of a clique in* $G^{\mathbf{A}}$ *equals to the objective value* V *of its corresponding feasible solution of the IQP problem for* $\mathbf{A}$.

Proof: Let $\mathcal{C}$ be a clique in $G^{\mathbf{A}}$ and $\mathbf{r}$ and $\mathbf{c}$ be the pair of vector of the corresponding feasible solution. For any user index i and packet index j, if $v_{ij} \in \mathcal{C}$, then $r_i = c_j = a_{ij} = 1$. Hence, every vertex in the clique adds 1 to V. ■

2) Reduction from Exact Cover by 3-Sets: Given a side information matrix A, the decision version of the IQP problem for $\mathbf{A}$, denoted as *D-IQP*, asks the following question: "Is there a feasible solution whose objective value equals N, for some $N > 0$?"

Proposition 7. *The D-IQP problem is NP-Complete.*

Proof: Clearly, D-IQP is in NP since given a feasible pair of vectors $\mathbf{r}$ and $\mathbf{c}$, we can compute the objective value in polynomial $O(nm)$ time.

In what follow, we show a reduction from the Exact Cover by 3-Sets (X3C) problem to D-IQP. X3C is well-known to be an NP-Complete problem [36] and is defined as follows:

Definition 2. *Given a set $\mathcal{E}$ of $3k$ elements: $\mathcal{E} = \{e_1, \cdots, e_{3k}\}$, and a collection $\mathcal{F} = \{\mathcal{S}_1, \cdots, \mathcal{S}_\ell\}$ of subsets $\mathcal{S}_i \subset \mathcal{E}$ and $|\mathcal{S}_i| = 3$, for $i \in [1, \ell], \ell > k$. The X3C problem asks the following question: "Are there k sets in $\mathcal{F}$ whose union is $\mathcal{E}$?"*

The reduction: Given any instance of X3C, we create $3k$ users, $u_1, \cdots, u_{3k}$, and ℓ packets, $p_1, \cdots, p_\ell$. The users correspond to the elements $e_i, i \in [1, 3k]$, and the packets correspond to the sets $S_j, j \in [1, \ell]$. We form the side information matrix $\mathbf{A}^{\text{X3C}}$ corresponding to this X3C instance by setting $a_{ij} = 1$ if $e_i \in \mathcal{S}_j$ and 0 otherwise.

Next, we will show that there is a feasible solution to the D-IQP for $\mathbf{A}^{\text{X3C}}$ whose objective value V equals $3k$ if and only if there are k sets $\mathcal{S}_{j_1}, \cdots, \mathcal{S}_{j_k}$ whose union is $\mathcal{E}$.

($\Rightarrow$) Let $\mathbf{r}$ and $\mathbf{c}$ be the pair of vectors of the feasible solution whose objective value $V = 3k$. First, observe that all r_i, for $i = 1, \cdots, 3k$, must equal 1; otherwise, assume there exists some index $t \in [1, 3k]$ where $r_t = 0$, then

$$V = \sum_{i=0}^{3k} r_i \sum_{j=0}^{\ell} c_j a_{ij} = \sum_{i=0, i\neq t}^{3k} r_i \sum_{j=0}^{\ell} c_j a_{ij} < 3k\,,$$

since each term $r_i \sum_{j=0}^{\ell} c_j a_{ij}$ is at most 1 by constraint (1). This is a contradiction.

Next, we create the corresponding solution to the X3C problem using $\mathbf{c}$. In particular, for $j = 1, \cdots, \ell$, we select S_j if $c_j = 1$. Because $V = 3k$ and $r_i = 1$ for all i, it must be that

$$\sum_{j=1}^{\ell} c_j a_{ij} = 1, \quad \text{for } i = 1, \cdots, 3k.$$

Thus, for a user index $s \in [1, 3k]$, there exists a *unique* packet index $t \in [1, \ell]$, where $c_t\, a_{st} = 1$, which means $c_t = a_{st} = 1$. By construction, we selected set S_t, and this S_t covers element s as $a_{st} = 1$. Therefore, every element is contained in exactly one set.

($\Leftarrow$) Let $\mathcal{S}_{j_1}, \cdots, \mathcal{S}_{j_k}$ be the solution to the X3C problem. We create the corresponding solution to the D-IQP problem as follows. First, for $\mathbf{r}$, let $r_i = 1$, for all $i = 1, \cdots, 3k$. Then, for $\mathbf{c}$, let $\mathcal{J} = \{j_1, \cdots, j_k\}$, and for $j = 1, \cdots, \ell$, set $c_j = 1$ if $j \in \mathcal{J}$ and 0 otherwise. Since $\mathcal{S}_{j_1}, \cdots, \mathcal{S}_{j_k}$ covers all $3k$ elements and each set has only 3 elements, each element e_s appears in exactly one set $\mathcal{S}_{j_t}$ for some $t \in [1, k]$, and $c_{j_t} = 1$. Thus, for each element $s \in [1, 3k]$,

$$\sum_{j=0}^{\ell} c_j a_{sj} = c_{j_t} a_{s\, j_t} = 1 \cdot 1 = 1$$

Given the above $\mathbf{r}$ and $\mathbf{s}$,

$$V = \sum_{i=0}^{3k} r_i \sum_{j=0}^{\ell} c_j a_{ij} = 3k \cdot 1 = 3k\,.$$

■

From Propositions 1, 4, and 7, we have the following main result of this work.

Theorem 8. *Given a side information matrix* $\mathbf{A}$ *and its IDNC graph* $G^{\mathbf{A}}$*, finding a maximum clique in* $G^{\mathbf{A}}$*, and equivalently, an optimal instantly decodable packet, is NP-Hard. Their corresponding decision versions are NP-Complete.*

V. Maximum Cliques in Random IDNC Graphs

In this section, we investigate Random Real-Time IDNC. In particular, we assume that each user, u_i, $i \in [1, n]$, fails to receive a packet, p_j, $j \in [1, m]$, with the same probability, $p \in (0, 1)$, independently. For ease of analysis, we assume that m is linear in n: $m = d\,n$, for some constant $d > 0$. (Our results also hold when m is polynomial in n.)

A random IDNC graph, denoted as $G^{\mathbf{A}}(p)$, is the graph corresponding to a side information matrix $\mathbf{A}$, whose each entry equals 1 with probability p and 0 with probability $q = 1 - p$ independently. Next, we will provide the analysis of the size of the maximum clique, *i.e.*, the clique number, of random IDNC graphs.

The main results of this section are the followings:

(i) For any $p \in (0, 1)$, the clique number for almost every graph in $G^{\mathbf{A}}(p)$ is linear in n. In particular, it equals $j^* p q^{j^*-1} n$, where $j^* = \text{argmax}\ j p q^{j-1}$, $j^* \in \mathbb{N}$. With high probability, the optimal recovery packet involves combining j^* packets.

(ii) With high probability, the maximum clique can be found in polynomial time, $O(n\,m^{j^*+\delta})$, where δ is a small constant parameter, and we provide an explicit algorithm, Max Clique, to find it. Consequently, the optimal recovery packet can be computed in polynomial time in n.

Comparison to Erdős-Rényi Random Graphs: Clique numbers of Erdős-Rényi random graphs with n vertices and $p = 1/2$ are known to be close to $2\log_2 n$ [37]. However, it is widely conjectured that for any constant $\epsilon > 0$, there does not exist a polynomial-time algorithm for finding cliques of size $(1 + \epsilon)\log_2 n$ with significant probability [38]. In contrast, for random

IDNC graphs with $n \times m$ vertices, where m is linear or polynomial in n, we show that the clique numbers are linear in n, and the corresponding cliques can be found in polynomial time in n.

A. Clique Number of Random IDNC Graphs

First, observe that any k 1's that lie in the same column form a clique of size k. Since the expected number of 1's in a single column is np, the expected size of single-column cliques is np. As a result, we expect the maximum clique size to be linear in n.

Fix a set $\mathcal{C}_j$ of j columns. A row r is said to be *good* with respect to $\mathcal{C}_j$ if among the j columns, it has 1 one and $j-1$ zeros. The probability that a row is good w.r.t. $\mathcal{C}_j$ is

$$f(j) = j\, p\, q^{j-1}\,. \tag{2}$$

Let $Z_{\mathcal{C}_j}$ be the number of good rows w.r.t $\mathcal{C}_j$. Then $Z_{\mathcal{C}_j}$ has a binomial distribution: $\text{Bin}(n, f(j))$.

Let $X_{\mathcal{C}_j}$ be the size of the maximum clique that has at least one vertex on every column in $\mathcal{C}_j$, *i.e.*, the clique touches j columns. Observe that if $j = 1$, then $f(1) = p$, and $X_{\mathcal{C}_1} = Z_{\mathcal{C}_1}$, which is the number of 1's in the chosen column. Thus, $X_{\mathcal{C}_1}$ has a Binomial distribution: $\text{Bin}(n, p)$. For $j > 1$, $X_{\mathcal{C}_j} \neq Z_{\mathcal{C}_j}$ since the set of good rows may not have a 1 in every column in $\mathcal{C}_j$. The following lemma states that for a large k, where $k \stackrel{def}{=} Z_{\mathcal{C}_j} \sim \text{Bin}(n, f(j))$, *i.e.*, given large enough n, $X_{\mathcal{C}_j} = Z_{\mathcal{C}_j}$ with high probability.

Lemma 9. *For a set of constant j columns $\mathcal{C}_j$, there exists a constant $k_j > 0$ such that for all $k \geq k_j$,*

$$\textit{Pr}[Z_{\mathcal{C}_j} = X_{\mathcal{C}_j} \mid Z_{\mathcal{C}_j} = k] \geq 1 - j\, \left(\frac{j-1}{j}\right)^k\,.$$

Proof: For $k \geq j > 0$, let B_k^j denote the number of ways to put k 1's into a matrix of size $k \times j$ such that (i) each row has one 1, and (ii) each column has at least one 1. Note that $B_k^1 = 1$, and we have the following recurrence:

$$B_k^j = j^k - \binom{j}{1} B_k^{j-1} - \binom{j}{2} B_k^{j-2} \cdots - \binom{j}{j-1} B_k^1\,. \tag{3}$$

This recurrence states that the number of ways to put k 1's into k rows (each row has one 1) using exactly j columns equals to the number of ways to put k 1's into k rows without any column restriction subtracts the cases where there are $1, 2, \cdots, j-1$ empty columns. It can be shown by induction (details are in Appendix C) that

$$B_k^j = \sum_{i=0}^{j-1} (-1)^i \binom{j}{i} (j-i)^k \,.$$

Thus, we have that

$$B_k^j = j^k - j(j-1)^k + \sum_{i=2}^{j-1} (-1)^i \binom{j}{i} (j-i)^k \,.$$

Let k_j be the minimum positive integer value of k such that $\sum_{i=2}^{j-1}(-1)^i \binom{j}{i}(j-i)^k \geq 0$. Then, for all $k \geq k_j$,

$$\Pr[Z_{\mathcal{C}_j} = X_{\mathcal{C}_j} \mid Z_{\mathcal{C}_j} = k] = \frac{B_k^j}{j^k} \geq \frac{j^k - j(j-1)^k}{j^k} \,.$$

■

The following lemma states that $X_{\mathcal{C}_j}$, the size of the maximum clique that touches all j columns, heavily concentrates around $nf(j)$ for large n.

Lemma 10. *For a set of constant j columns $\mathcal{C}_j$ and any constant $c > 1$, let $\mu = nf(j)$ and $\delta = \sqrt{\frac{3c \ln n}{n f(j)}}$. For a large n such that $\mu - \mu\delta \geq k_j$ (k_j is as in Lemma 9), we have*

$$Pr[\,|\, X_{\mathcal{C}_j} - \mu \,|\, \geq \mu\delta] \leq \frac{2}{n^c} + 2\mu\delta\, j(1 - \frac{1}{j})^{\mu - \mu\delta} \,.$$

This probability goes to 0 as $n \to \infty$.

The proof is provided in Appendix D. Intuitively, this result follows from $X_{\mathcal{C}_j} = Z_{\mathcal{C}_j}$ w.h.p. (Lemma 9), and the fact that the Binomial distributed $Z_{\mathcal{C}_j}$, the number of good rows, concentrates heavily around its mean, $nf(j)$. Note that $\mu\delta$ is $\Theta(\sqrt{n \ln n})$; thus, $X_{\mathcal{C}_j}$ is within $\Theta(\sqrt{n \ln n})$ of $nf(j)$ w.h.p.

Next, for a constant j, let X_j be the size of the maximum clique that touches *any* j columns. X_j also heavily concentrates around $nf(j)$. Recall that $m = dn$, for some constant $d > 0$. Formally,

Theorem 11. *For a constant j and any constant $c > j$, let $\mu = nf(j)$ and $\delta = \sqrt{\frac{3c \ln n}{n f(j)}}$. For a large n such that $\mu - \mu\delta \geq k_j$ (k_j is as in Lemma 9), we have*

$$Pr[\,|\, X_j - \mu \,|\, \geq \mu\delta] \leq \frac{2d^j}{n^{c-j}} + 2d^j n^j \mu\delta\, j(1 - \frac{1}{j})^{\mu - \mu\delta} \,.$$

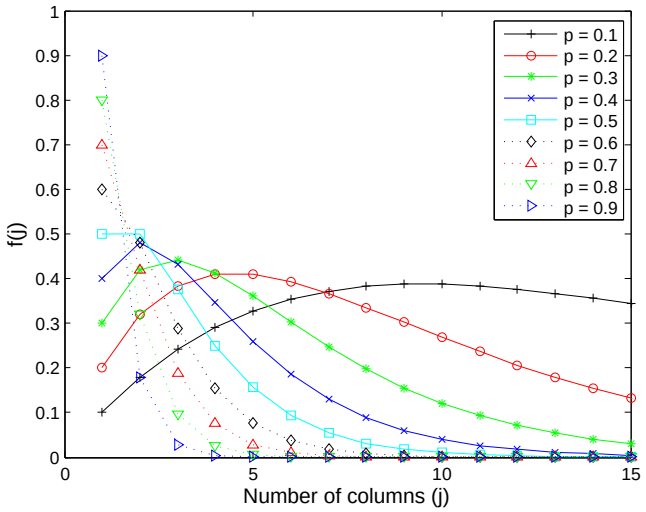


Fig. 3. Plot of $f(j) = jp(1-p)^{j-1}$ for different loss rate p

This probability goes to 0 as $n \to \infty$.

Proof: The proof is by using the union bound on the result of Lemma 10:

$$\begin{aligned}
\Pr[\,|\,X_j - \mu\,|\,\geq \mu\delta] &= \Pr[\cup_{\mathcal{C}_j}\,|\,X_{\mathcal{C}_j} - \mu\,|\,\geq \mu\delta] \\
&\leq \binom{m}{j}\Pr[\,|\,X_{\mathcal{C}_j} - \mu\,|\,\geq \mu\delta] \\
&\leq m^j\left(\frac{2}{n^c} + 2\mu\delta\, j(1-1/j)^{\mu-\mu\delta}\right) \\
&\leq \frac{2d^j}{n^{c-j}} + 2d^j n^j \mu\delta\, j(1-1/j)^{\mu-\mu\delta}\,.
\end{aligned}$$

■

We note that the above concentration result also holds when the number of packets, m, is polynomial in the number of user, n, *i.e.*, $m = n^d$, for some constant $d > 0$. However, it needs a larger constant c ($c > d\,j$), which means less concentration (as δ is larger). Apparently, the results do not hold when m is exponential in n. However, the cases where m is either linear or polynomial in n are sufficient for practical purposes as in real-time applications, such as [7], m is linear in n.

Now let $j^* = \text{argmax} f(j), j^* \in \mathbb{N}$. There may be a set of consecutive values of $j \in \mathbb{N}$ that maximize $f(j)$, in that case, pick j^* to be the smallest value among them. Note that for a

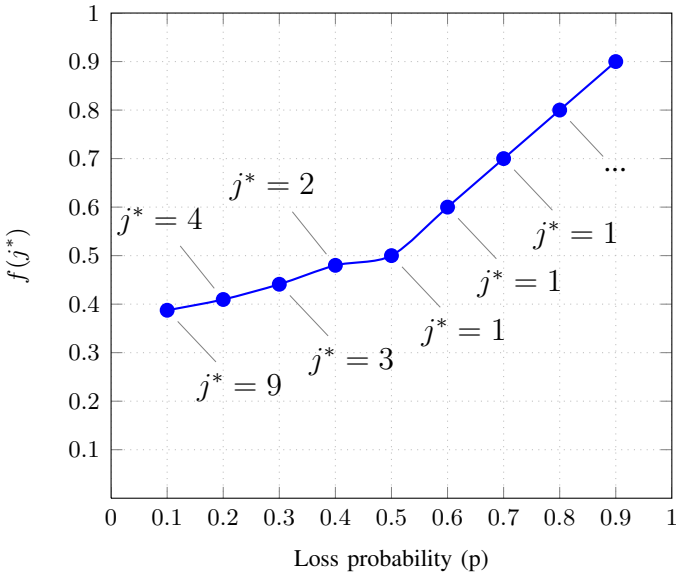


Fig. 4. Values of $f(j^*)$ and its corresponding j^*. The clique number heavily concentrates around $f(j^*) \times n$, and j^* is the number of packets should be coded together.

constant p, j^* and $f(j^*)$ are also constant.

Corollary 12. *For a sufficiently large n, with high probability, the maximum clique touches a constant number j^* of columns, where $j^* = argmax f(j)$.*

Proof: Intuitively, this follows from the above result that the size of the maximum clique that touches j columns heavily concentrates around $nf(j)$. In detail, for any constant j' such that $f(j') < f(j^*)$, let $c > \max(j', j^*) + 1$. Theorem 11 implies that w.h.p., the size of the maximum clique that touches any j' column is at most

$$k' = nf(j') + \sqrt{3cf(j')n \ln n}\,,$$

and the size of the maximum clique that touches any j^* column is at least

$$k'' = nf(j^*) - \sqrt{3cf(j^*)n \ln n}\,.$$

For a large enough n, it is clear that $k' < k''$. ■

Fig. 3 plots the function $f(j)$ for different values of p. This plot shows that (i) for $p >= 0.5$, $f(j)$ is a decreasing function, and for $p < 0.5$, $f(j)$ initially increases then decreases, and (ii) j^* increases as p decreases, which suggests the following result:

The number of packets should be coded together increases when the loss rate decreases.

Fig. 4 plots the values of $f(j^*)$ and the corresponding values of j^*. An important observation from Fig. 4 is that even when the loss rate is small, the clique size is still high. For instance, when $p = 0.1$, we have $j^* = 9$ and $f(j^*) \simeq 0.38$, which means that the optimal coded packet involves coding 9 plain packets together, and this packet will benefit about 38% of the users.

B. Finding a Maximum Clique

Based on the analysis in the previous section, we propose Max Clique (Algorithm 1) to find a maximum clique of a given random IDNC graph. Max Clique examines all cliques that touch j columns, for all j combinations of m columns, where j is within a small constant δ neighborhood of j^*. In the case j^* is larger than m, j^* is set to m (Line 1), exploiting the fact that for $j < j^*$, $f(j)$ is an increasing function as shown in Fig. 3.

Complexity. In Max Clique, the for each loop starting at Line 3 runs at most $2\delta\binom{m}{j^*+\delta}$ times. The for loop starting at Line 5 runs n times. The if condition check at Line 6 examines up to $j^*+\delta$ entries. Thus, the total runtime of Algorithm 1 is at most $2\delta\binom{m}{j^*+\delta}n(j^*+\delta) = O(n\,m^{j^*+\delta})$, which is polynomial in n when m is linear or polynomial in n.

Optimal Coded Packet. Given the vertices of the maximum clique output by Max Clique, one can readily compute the optimal instantly decodable packet by XORing the packets whose indices correspond to the packet indices of the output vertices, as indicated in Proposition 1.

VI. Performance Evaluation

A. Numerical Evaluation

In this section, we use simulation to compare the performance of the proposed Max Clique algorithm (Algorithm 1) against two baselines proposed in [2]: an optimal repetition-based algorithm, called Best Repetition and a COPE-like greedy-based algorithm.

The *Best Repetition* algorithm rebroadcasts the plain packet that is wanted by the most number of users. This is inherently the best repetition strategy. The *COPE-Like* algorithm goes through all the packets that are still wanted by at least one user in a random order, and it tries to compute a coded packet that is instantly decodable to *all* users. In particular, it begins by selecting the first packet of a random permutation, $c = p_1$. It then goes through the rest of the packets one by one. At each step j, $j > 1$, it XORs the packet p_j under consideration with c: $c = c \oplus p_j$, if

Algorithm 1 *Max Clique*: Finding the Maximum Clique

Input: p: loss probability, n: number of users, m: number of packets, $\mathbf{A}$: side information matrix of size $n \times m$.

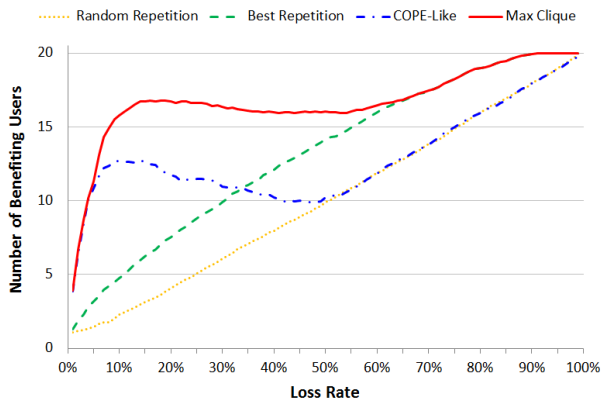
Output: $\mathcal{I}^*$: vertices of the maximum clique

$j^* \leftarrow \min(m, \text{argmax}_{j \in \mathbb{N}} f(j))$
$\mathcal{I}^* = \emptyset$
for each combination of j columns out of m columns, where $j \in [j^* - \delta, j^* + \delta]$
    $\mathcal{I} = \emptyset$
    for $r = 1 \rightarrow n$ **do**
        if row r has only one 1 at column c **then**
            Add (r, c) to $\mathcal{I}$
        end if
    end for
    if $|\mathcal{I}| > |\mathcal{I}^*|$ **then**
        $\mathcal{I}^* = \mathcal{I}$
    end if
end

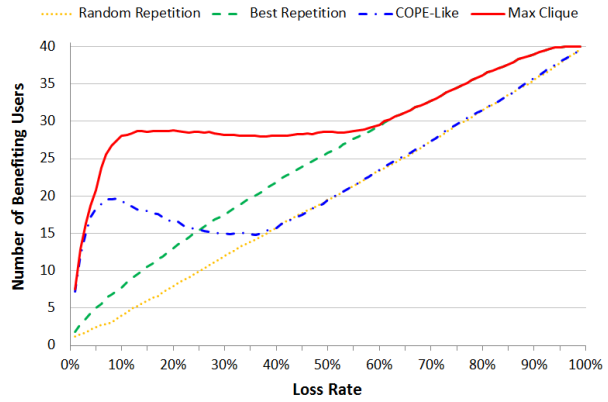
the result is still instantly decodable to all users; otherwise, it skips p_j. For reference, we also include the Random Repetition algorithm, which resends a random packet that is still wanted by at least one user.

Settings. For each loss rate ranging from 1% to 99%, per 1% increment, we randomly generate 100 side information matrices. We then run the algorithms on these matrices. For the Max Clique algorithm, we set δ, the neighborhood around j^*, to 3. Fig. 5 plots the average numbers of beneficiary users as a function of loss rate for the two parameter settings $\{n = 20, m = 20\}$ and $\{n = 40, m = 20\}$. For clarity, we skip plotting the standard deviations: they are ranging from 0 to 3 for all algorithms.

Results. In Fig. 5, we can see that the proposed Max Clique algorithm consistently and significantly outperforms all other algorithms. In particular, for the case $\{n = 20, m = 20\}$, on average, Max Clique performs 1.3 times better than both the Best Repetition and COPE-Like.



(a) $n = 20$ users, $m = 20$ packets



(b) $n = 40$ users, $m = 20$ packets

Fig. 5. Performance of the proposed Max Clique coding scheme in comparison with those of the Best Repetition and COPE-Like coding schemes.

For the loss rates between 40% and 50%, Max Clique performs up to 1.6 times better than the COPE-Like algorithm, and for the loss rates between 10% and 15%, Max Clique performs up to 3.8 times better than the Best Repetition algorithm. Similar trend but higher improvement, 1.35 times on average and up to 4.5 times, could be observed for the case $\{n = 40, m = 20\}$ in Fig 5(b).

Two interesting observations can be made from Fig. 5(a) (and similarly for Fig. 5(b)): (i) When the loss rate is larger than a certain threshold (65% in Fig. 5(a)), the performance of Max Clique is similar to that of the Best Repetition, which suggests that Max Clique also tries to select the best uncoded packet. This is because a plain packet now benefits many users due to high loss rate. (ii) When the loss rate is larger than another threshold (50% in Fig. 5(a)), the

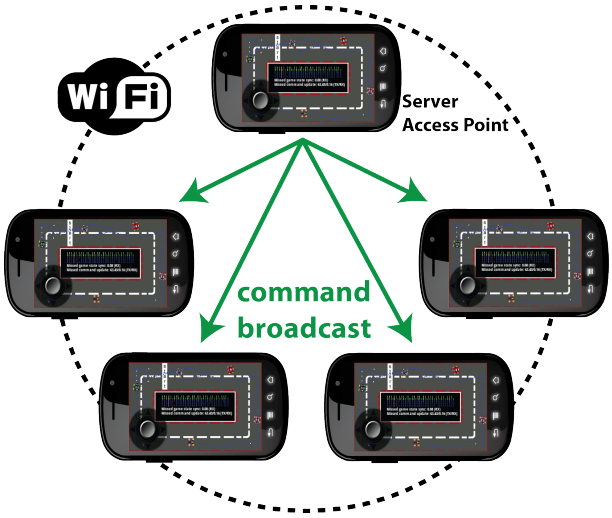


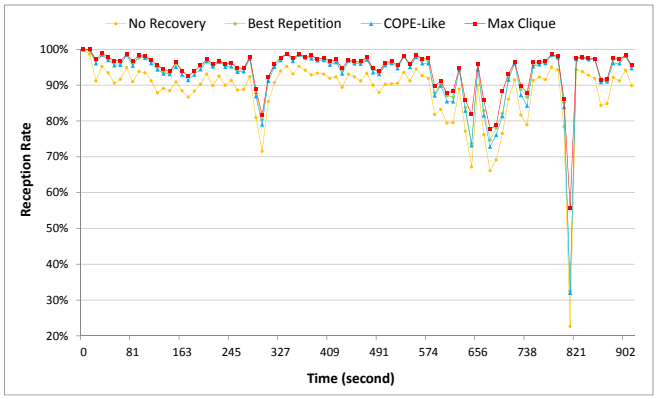
Fig. 6. MicroPlay networking model: One phone acts as the WiFi access point and as the game server. This phone uses WiFi broadcast to disseminate its game commands.

performance of COPE-Like is similar to that of Random Repetition, which suggests that packets cannot be coded together while being instantly decodable to all users. This is because when the loss rate is high, given any pair of 2 plain packets, there exists a user who lost both w.h.p.

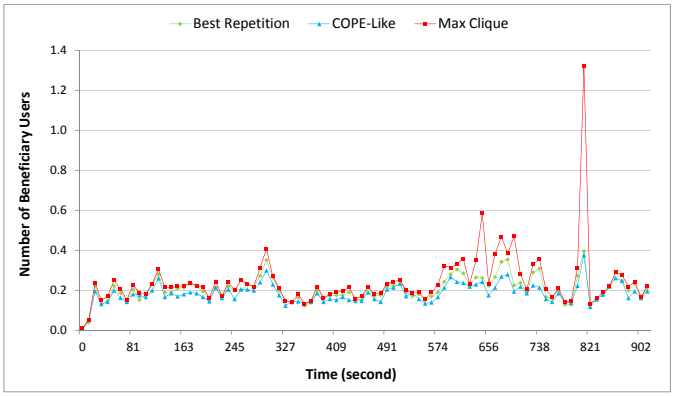
B. Trace-Based Evaluation

In this section, we evaluate the performance of Max Clique in comparison with the baselines, using real network traces of an Android application called Racer [7]. Racer is a real-time multi-player racing game implemented on top of a networking framework, called MicroPlay, that we previously developed [7]. MicroPlay exploits wireless broadcast to disseminate input commands from one player to the rest in a timely manner to support accurate game rendering and low latency.

In particular, in Racer, each player's car races around a closed rectangular track and broadcast its movement continuously to the rest of the players. A player uses the broadcast packets to update the positions of the other players' cars. In the context of this work, we examine the packets broadcast by one player, who is acting as the game server and the WiFi access point to the group, depicted in Fig. 6. This scenario we select for evaluation here, in principle, matches the broadcast scenario that we examined earlier in our analysis in Fig. 1.



(a) Reception rate



(b) Number of beneficiary users

Fig. 7. Trace-based performance of Max Clique in comparison with the baselines when 1 recovery packet is broadcast per 10 packets. The recovery packet is assumed to be received successfully.

Trace Collection and Description. We created a Racer game session that has 5 players: 1 server and 4 clients, as shown in Fig. 6. The hardware in use consist of 3 Samsung Captivate and 2 Nexus S phones, all running Android OS 2.3 (Gingerbread). The players are scattered in an on-campus cafeteria, whose area is of sizes approximately 40 x 40 meters. The game session

occurs during a busy lunch hour[1].

Each packet broadcast by the server has a unique identification number. We implemented a statistics-collection software module within the Racer game client to capture the reception of the packets broadcast by the server. In particular, each client logs the packets it were able to receive and the time it received them. The game session lasted about 15 minutes, and during the game, the server broadcast 19,059 packets, about a packet every 47 ms on average.

The average reception rate of all 4 clients during the game is shown in Fig. 7(a) by the "No Recovery" line. Each point plotted represents the average reception rate of packets broadcast within a 10-second bin. Fig. 7(a) shows that the average reception rate of the clients is high: most of the time above 90%. Nevertheless, there are several instances when the average reception rate drops below 90%, for example, from second 574 to 738. Also, the average reception rate drops as low as 23% at second 811. The reception rates are quite similar across the clients. For this reason, we skip reporting the plots of the individual client rates.

Settings. For each batch of packet of size B, we compute a recovery packet using the Best Repetition, COPE-Like, and Max Clique algorithms. This recovery packet is to be broadcast at the end of each batch by the server to recover packet losses at the client. For evaluation purposes, we assume that this packet would be successfully received by all the clients. We then compute the new reception rates at the clients for each recovery scheme.

Results. Fig. 7(a) plots the average reception rate when each of the recovery schemes is used for batch size $B = 10$. It could be observed from this figure that Max Clique consistently outperforms the COPE-Like and Best Repetition. In other words, the improvement of the average reception rate is higher when Max Clique is used to compute the recovery packet.

In more details, Fig. 7(b) plots the number of beneficiary users when each of the recovery scheme is used. Each point plotted is the average over multiple recovery packets within a 10-second bin. Fig. 7(b) shows that the recovery packets computed by Max Clique consistently benefit more users: on average, Max Clique helps 16% more users than Best Repetition and 26% more users than COPE-Like. The performance gaps between Max Clique and the baselines are more noticeable when the reception rates are low, *e.g.*, between second 574 and 738, or at

[1]We also capture multiple network traces in other hours. We report here the representative traces.

second 811, where Max Clique helps 50–250% more users than the others.

We also perform similar evaluation for batches of sizes $B = 5$ and $B = 20$. For $B = 5$, the average performance improvement of Max Clique over Best Repetition is 5% and over COPE-Like is 12%, which are less than those when $B = 10$. This is due to the reduced number of coding opportunities (over just 5 packets). For $B = 20$, the average performance improvement of Max Clique over Best Repetition is 12% and over COPE-Like is 28%, which are similar to those when $B = 10$. This implies that $B = 10$ creates sufficient coding opportunities for the loss rates of this set of traces.

Finally, unlike the numerical results reported in the previous section, Fig. 7 shows that Best Repetition consistently outperforms COPE-Like. This is likely due to the dependency of the packet losses at the clients: a packet lost at a client is likely to be lost at other clients, which implies that re-sending this packet might benefit many clients. This also occurs when $B = 5$ and $B = 20$.

VII. Conclusion

In this paper, we formulate the Real-Time IDNC problem, which seeks to compute a recovery packet that is immediately beneficial to the maximum number of users. Our analysis shows that Real-Time IDNC is NP-Hard. We then analyze the Random Real-Time IDNC, where each user is assumed to lose every packet with the same probability independently. When the number of packets is linear or polynomial in the number of users, we show that the optimal packet could be computed in polynomial time in the number of users w.h.p., and we provide an explicit algorithm to find the optimal packet. We evaluate the proposed algorithm numerically as well as experimentally based on real network traces. The results of the evaluation confirm the superior performance of the proposed algorithm. In the future, we plan to extend this work from a single recovery time slot to a constant number of time slots, corresponding to larger delay tolerance.

Appendix A
Proof of Lemma 2

Proof:

($\Leftarrow$) We first show that a clique in $G^{\mathbf{A}}$ maps to an instantly decodable packet, which is uniquely identified by the user and packet indices of the vertices in the clique.

Let $\mathcal{C}$ be a clique in $G^{\mathbf{A}}$: $\mathcal{C} = \{v_{i_1,j_1}, \cdots, v_{i_k,j_k}\}$. Without loss of generality, assume $j_1, \cdots, j_k$ are pair-wise distinct, compute $c = p_{j_1} \oplus \cdots \oplus p_{j_k}$. (If $j_{t_1} = j_{t_2} = \cdots = j_{t_n}, n > 1$ then include only j_{t_1} in the XOR.) c is an instantly decodable packet with respect to the set of users $\{u_{i_1}, \cdots, u_{i_k}\}$ because

- For any user u_{i_t}, for some $t \in [1, k]$, the existence of vertex v_{i_t,j_t} indicates that it wants p_{j_t}. In the following, we show that u_{i_t} can decode for p_{j_t} immediately upon receiving c. Without loss of generality, consider user u_{i_1}. It suffices to show that u_{i_1} has all other packets in c. To see this, assume otherwise, *i.e.*, assume u_{i_1} does not have packet p_{j_s}, for some $s \in [2, k]$ where $p_{j_s} \neq p_{j_1}$. Then there is no edge between v_{i_1,j_1} and v_{i_s,j_s}. (contradiction)
- Each component, $p_{j_t}, t \in [1, k]$, of c is wanted by u_{i_t}.

($\Rightarrow$) We now show that an instantly decodable packet maps to a clique in $G^{\mathbf{A}}$, which is uniquely identified by the packets involved and the set of beneficiary users. Let $c^{\mathcal{N}} = p_{j_1} \oplus \cdots \oplus p_{j_k}$ be an instantly decodable packet with respect to the set of user $\mathcal{N}$. Let p_{j_t} be wanted by distinct users $\{u_1^{j_t}, \cdots, u_{n_t}^{j_t}\}$, for some $n_t > 0$. The following set of vertices, $\mathcal{C}$, form a clique in $G^{\mathbf{A}}$:

$$\mathcal{C} = \left\{v_{u_1^{j_1},j_1}, \cdots, v_{u_{n_1}^{j_1},j_1}, \cdots\cdots, v_{u_1^{j_k},j_k}, \cdots, v_{u_{n_k}^{j_k},j_k}\right\}.$$

We will show that there is an edge between any two vertices in C:

- For any $t \in [1, k]$, consider any pair $a \neq b, a, b \in [1, n_t]$. There is an edge of type (i) between $v_{u_a^{j_t},j_t}$ and $v_{u_b^{j_t},j_t}$ since both $u_a^{j_t}$ and $u_a^{j_t}$ need p_{j_t}.
- For any pair of $s \neq t, s, t \in [1, k]$, consider $a \in [1, n_s]$ and $b \in [1, n_t]$. There is an edge of type (ii) between $v_{u_a^{j_s},j_s}$ and $v_{u_b^{j_t},j_t}$. This is because $u_a^{j_s}$ must have p_{j_t} as it can decode for p_{j_s} immediately, and $u_b^{j_t}$ must have p_{j_s} as it can decode for p_{j_t} immediately.

Finally, it is easy to check that for the above two mappings, one is the reverse mapping of the other. ■

APPENDIX B
PROOF OF LEMMA 3

Proof: Let $c^{\mathcal{N}} = p_{j_1} \oplus \cdots \oplus p_{j_k}$ be an instantly decodable packet w.r.t. the set of user $\mathcal{N}$. Let p_{j_t}, $t \in [1, k]$, benefits distinct users $\{u_1^{j_t}, \cdots, u_{n_t}^{j_t}\}$, for some $n_t > 0$. The following set of

vertices, $\mathcal{C}$, forms the clique corresponding to $c^{\mathcal{N}}$:

$$\mathcal{C} = \left\{ v_{u_1^{j_1},j_1}, \cdots, v_{u_{n_1}^{j_1},j_1}, \cdots\cdots, v_{u_1^{j_k},j_k}, \cdots, v_{u_{n_k}^{j_k},j_k} \right\}.$$

To show $|\mathcal{N}| = |\mathcal{C}|$, it suffices to show that all user indices of vertices in $\mathcal{C}$ are pair-wise distinct. For any pair of $s \neq t$, where $s, t \in [1, k]$, consider any $a \in [1, n_s]$ and any $b \in [1, n_t]$. $u_a^{j_s} \neq u_b^{j_t}$ because otherwise $u_a^{j_s}$ cannot decode for p^{j_s}. ■

APPENDIX C
PROOF OF LEMMA 9 RECURRENCE

Proof: It can be shown by induction that

$$B_k^j = \sum_{i=0}^{j-1} (-1)^i \binom{j}{i} (j-i)^k . \tag{4}$$

In detail, assume that (4) is true for all indices $1, 2, \cdots, j-1$, then following from recurrence (3),

$$\begin{aligned} B_k^j &= j^k - \binom{j}{1} B_k^{j-1} - \binom{j}{2} B_k^{j-2} \cdots - \binom{j}{j-1} B_k^1 \\ &= j^k - \binom{j}{1} \sum_{i=0}^{j-2} (-1)^i \binom{j-1}{i} (j-1-i)^k \\ &\quad - \binom{j}{2} \sum_{i=0}^{j-3} (-1)^i \binom{j-2}{i} (j-2-i)^k \\ &\quad - \cdots \\ &\quad - \binom{j}{j-1} \cdot 1 . \end{aligned}$$

Now, for any $t \in [1, j-1]$, the coefficient of $(j-t)^k$ is

$$\sum_{i=1}^{t} \binom{j}{i} \binom{j-i}{t-i} (-1)^{t-i+1} .$$

Thus, it suffices to show that, for any $t \in [1, j-1]$,

$$\sum_{i=1}^{t} \binom{j}{i} \binom{j-i}{t-i} (-1)^{t-i+1} = (-1)^t \binom{j}{t} .$$

The above equation holds iff

$$\sum_{i=1}^{t} \frac{(-1)^{t-i+1}}{i!\,(t-i)!} = \frac{(-1)^t}{t!} .$$

Or, equivalently

$$\sum_{i=1}^{t} \binom{t}{i} (-1)^{t-i} = (-1)^{t+1} .$$

The LHS of the above equation equals

$$-(-1)^{t} + \sum_{i=0}^{t} \binom{t}{i} (-1)^{t-i} = (-1)^{t+1} ,$$

where the last "=" follows from the binomial theorem (for $a = 1, b = -1$). ■

APPENDIX D
PROOF OF LEMMA 10

Proof: Denote $Z_{\mathcal{C}_j}$ and $X_{\mathcal{C}_j}$ by Z and X, respectively. Applying Chernoff's bound on the Binomial distributed variable Z, we have

$$\Pr[\,|\,Z - \mu\,|\, \geq \mu\delta] \leq 2\exp(-\frac{\mu\delta^2}{3}) = \frac{2}{n^c} .$$

Now,

$$\begin{aligned}
&\Pr[\,|\,X - \mu\,|\, \geq \mu\delta] \\
&= \sum_{k=1}^{n} \Pr[\,|\,X - \mu\,|\, \geq \mu\delta \,|\, Z = k] \cdot \Pr[Z = k] \\
&\leq \sum_{k=1}^{\mu-\mu\delta} \Pr[Z = k] + \sum_{k=\mu+\mu\delta}^{n} \Pr[Z = k] \\
&\quad + \sum_{k=\mu-\mu\delta+1}^{\mu+\mu\delta-1} \Pr[\,|\,X - \mu\,|\, \geq \mu\delta \,|\, Z = k] \\
&\leq \frac{2}{n^c} + \sum_{k=\mu-\mu\delta+1}^{\mu+\mu\delta-1} \Pr[\,|\,X - \mu\,|\, \geq \mu\delta \,|\, Z = k]
\end{aligned}$$

For $Z \in [\mu - \mu\delta + 1, \mu + \mu\delta - 1]$, we have

$$\begin{aligned}
&\Pr[\,|\,X - \mu\,|\, \geq \mu\delta] \\
&= \Pr[\,|\,X - \mu\,|\, \geq \mu\delta, X = Z] \cdot \Pr[X = Z] \\
&\quad + \Pr[\,|\,X - \mu\,|\, \geq \mu\delta, X \neq Z] \cdot \Pr[X \neq Z] \\
&\leq \Pr[\,|\,X - \mu\,|\, \geq \mu\delta, X = Z] + \Pr[X \neq Z] \\
&= 0 + \Pr[X \neq Z] \\
&\leq j(1 - 1/j)^{\mu - \mu\delta} \quad \text{(from Lemma 9)}
\end{aligned}$$

Thus,

$$\Pr[\,|\,X - \mu\,|\, \geq \mu\delta] \leq \frac{2}{n^c} + 2\mu\delta\, j(1 - 1/j)^{\mu - \mu\delta}\,.$$

■

REFERENCES

[1] D. Nguyen, T. Tran, T. Nguyen, and B. Bose, "Wireless Broadcast Using Network Coding, *IEEE Transactions on Vehicular Technology*, vol. 58, no. 2, pp. 914–925, Feb. 2009.

[2] L. Keller, E. Drinea, and C. Fragouli, "Online Broadcasting with Network Coding," in *Proceedings of IEEE International Symposium on Network Coding (NetCod) 2008*, Hong Kong, pp. 1–6.

[3] P. Sadeghi, D. Traskov, and R. Koetter, "Adaptive network coding for broadcast channels," in *Proceedings of IEEE International Symposium on Network Coding (NetCod) 2009*, Lausanne, pp. 80–85.

[4] S. Sorour and S. Valaee, "On Minimizing Broadcast Completion Delay for Instantly Decodable Network Coding," in *Proceedings of IEEE International Conference on Communications (ICC) 2010*, Cape Town, pp. 1–5.

[5] M. Luby, M. Watson, T. Gasiba, T. Stockhammer, W. Xu, "Raptor codes for reliable download delivery in wireless broadcast systems," in *Proceedings of IEEE Consumer Communications and Networking Conference (CCNC) 2006*, Las Vegas, pp. 192–197.

[6] J. W. Byers, M. Luby, M. Mitzenmacher, A. Rege, "A digital fountain approach to reliable distribution of bulk data," in *Proceedings of ACM SIGCOMM 1998*, Vancouver, pp. 56–67.

[7] A. Le, L. Keller, C. Fragouli, and A. Markopoulou, "MicroPlay: A Networking Framework for Local Multiplayer Games," in *Proceedings of ACM SIGCOMM Workshop on Mobile Gaming (MobiGames) 2012*, Helsinki, pp.155–160.

[8] S. Sorour and S. Valaee, "Minimum Broadcast Decoding Delay for Generalized Instantly Decodable Network Coding," in *Proceedings of IEEE Global Communications Conference (Globecom) 2010*, Miami, pp. 1–5.

[9] S. Sorour and S. Valaee, "Completion Delay Minimization for Instantly Decodable Network Coding with Limited Feedback," in *Proceedings of IEEE International Conference on Communications (ICC) 2011*, Kyoto, pp. 1–5.

[10] S. Sorour and S. Valaee, "Completion delay reduction in lossy feedback scenarios for instantly decodable network coding," in *Proceedings of IEEE International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC) 2011*, Toronto, pp. 2025–2029.

[11] S. Katti, H. Rahul, W. Hu; D. Katabi, M. Medard, and J. Crowcroft, "XORs in the Air: Practical Wireless Network Coding," in *IEEE/ACM Transactions on Networking (ToN)*, vol. 16, no. 3, pp. 497–510, Jun. 2008.

[12] S. Sorour and S. Valaee, "Adaptive network coded retransmission scheme for wireless multicast," in *Proceedings of IEEE International Symposium on Information Theory (ISIT) 2009*, Seoul, pp. 2577–2581.

[13] S. Sorour and S. Valaee, "On densifying coding opportunities in instantly decodable network coding graphs," in *Proceedings of IEEE International Symposium on Information Theory (ISIT) 2012*, Boston, pp. 2456–2460.

[14] N. Aboutorab, S. Sorour, P. Sadeghi, "O2-GIDNC: Beyond Instantly Decodable Network Coding," in *Proceedings of IEEE International Symposium on Network Coding (NetCod) 2013*, Calgary, pp. 1–6.

[15] X. Li, C. Wang, and X. Lin, "On The Capacity of Immediately-Decodable Coding Schemes for Wireless Stored-Video Broadcast with Hard Deadline Constraints," in *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 29, no. 5, pp. 1094–1105, May 2011.

[16] Y. Birk and T. Kol, "Coding-on-demand by an informed source (ISCOD) for efficient broadcast of different supplemental data to caching clients," in *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2825–2830, Jun. 2006.

[17] S. El Rouayheb, M. A. R. Chaudhry, and A. Sprintson, "On the Minimum Number of Transmissions in Single-Hop Wireless Coding Networks," in *Proceedings of IEEE Information Theory Workshop (ITW) 2007*, Lake Tahoe, pp. 120–125.

[18] H. Maleki, V. Cadambe, and S. Jafar, "Index Coding: An Interference Alignment Perspective", in *Proceedings of IEEE International Symposium on Information Theory (ISIT) 2012*, Boston, pp. 2236–2240.

[19] A. Saber Tehrani, and A. G. Diamkis, "Finding Three Transmissions is Hard," in *Proceedings of IEEE Global Communications Conference (Globecom) 2012*, Anaheim, pp. 2317–2322.

[20] S. Brahma, C. Fragouli, "Pliable Index Coding," in *Proceedings of IEEE International Symposium on Information Theory (ISIT) 2012*, Boston, pp. 2251–2255.

[21] S. Brahma, C. Fragouli, "Pliable Index Coding: The multiple requests case," in *Proceedings of IEEE International Symposium on Information Theory (ISIT) 2013*, Istanbul, pp. 1142–1146.

[22] M. Langberg and A. Sprintson, "On the hardness of approximating the network coding capacity," in Proceedings of IEEE International Symposium on Information Theory (ISIT) 2008, Toronto, pp. 315319.

[23] M.A.R. Chaudhry and A. Sprintson, "Efficient algorithms for Index Coding," in *Proceedings of IEEE International Conference on Computer Communications (INFOCOM) Workshops 2010*, Phoenix, pp. 1–4.

[24] A. Saber Tehrani, A. G. Diamkis, and M. J. Neely, "Bipartite Index Coding," in *Proceedings of IEEE International Symposium on Information Theory (ISIT) 2012*, Boston, pp. 2246–2250.

[25] S. El Rouayheb, A. Sprintson, and P. Sadeghi, "On Coding for Cooperative Data Exchange," in *Proceedings of IEEE Information Theory Workshop (ITW) 2010*, Cairo, pp. 1–5.

[26] S. El Rouayheb, P. Sadeghi, and A. Sprintson, "A Randomized Algorithm and Performance Bounds for Coded Cooperative Data Exchange," in *Proceedings of IEEE International Symposium on Information Theory (ISIT) 2010*, Austin, pp. 1888–1892.

[27] N. Milosavljevic, S. Pawar, S. El Rouayheb, M. Gastpar, and K. Ramchandran, "Deterministic Algorithm for the Cooperative Data Exchange Problem," in *Proceedings of IEEE International Symposium on Information Theory (ISIT) 2011*, St Petersburg, pp. 410–414.

[28] N. Milosavljevic, S. Pawar, S. El Rouayheb, M. Gastpar, and K. Ramchandran, "Data Exchange Problem with Helpers," in *Proceedings of IEEE International Symposium on Information Theory (ISIT) 2012*, Boston, pp. 2611–2615.

[29] A. Sprintson, P. Sadeghi, G. Booker, and S. El Rouayheb, "Deterministic Algorithm for Coded Cooperative Data Exchange,"

in *Proceedings of ICST Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness (QShine) 2010*, Houston.

[30] D. Ozgul and A. Sprintson, "An Algorithm for Cooperative Data Exchange with Cost Criterion," in *Proceedings of Information Theory and Applications Workshop (ITA) 2011*, San Diego, pp. 1–4.

[31] M. Gonen and M. Langberg, "Coded Cooperative Data Exchange Problem for General Topologies," in *Proceedings of IEEE International Symposium on Information Theory (ISIT) 2012*, Boston, pp. 2606–2610.

[32] T.A. Courtade, X. Bike, and R.D. Wesel, "Optimal exchange of packets for universal recovery in broadcast networks," in *Proceedings of Military Communications Conference (MILCOM) 2010*, San Jose, pp. 2250–2255.

[33] T.A. Courtade and R.D. Wesel, "Efficient universal recovery in broadcast networks," in *Proceedings of Allerton Conference on Communication, Control and Computing (Allerton) 2010*, Urbana-Champaign, pp. 1542–1549.

[34] T.A. Courtade and R.D. Wesel, "Weighted universal recovery, practical secrecy, and an efficient algorithm for solving both," in *Proceedings of Allerton Conference on Communication, Control and Computing (Allerton) 2011*, Urbana-Champaign, pp. 1349–1357.

[35] A. Le, A. S. Tehrani, A. G. Dimakis, A. Markopoulou, "Instantly Decodable Network Codes for Real-Time Application," in *Proceedings of IEEE International Symposium on Network Coding (NetCod) 2013*, Calgary, pp. 1–6.

[36] M. R. Garey and D. S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-Completeness," W. H. Freeman & Co. New York, NY, 1979.

[37] N. Alon and J. Spencer, "The Probabilistic Method," Wiley, New York, 1992.

[38] A. Juels and M. Peinado, "Hiding Cliques for Cryptographic Security," in *Proceedings of ACM-SIAM Symposium on Discrete Algorithms '98*, San Francisco, pp. 678–684.