On the Hardness of Approximating the Network Coding Capacity

Michael Langberg
Computer Science Division
The Open University of Israel
Raanana, 43107, Israel
mikel@openu.ac.il

Alex Sprintson
Department of Electrical and Computer Engineering
Texas A&M University
College Station, Texas, 77843
spalex@ece.tamu.edu

***Abstract*—This work addresses the computational complexity of achieving the capacity of a general network coding instance. We focus on the linear capacity, namely the capacity of the given instance when restricted to linear encoding functions. It has been shown [Lehman and Lehman SODA 2005] that determining the (scalar) linear capacity of a general network coding instance is NP-hard. In this work we initiate the study of *approximation* in this context. Namely, we show that given an instance to the general network coding problem of linear capacity C, constructing a linear code with rate αC for any universal constant $\alpha \leq 1$ (independent of the size of the instance) is "hard". Specifically, finding such network codes would solve a long standing open problem in the field of graph coloring.**

In addition, we consider the problem of determining the (scalar) linear capacity of a *planar* network coding instance (a general instance in which the underlying graph is planar). We show that even for planar networks this problem remains NP-hard.

I. Introduction

In the network coding paradigm, internal nodes of the network may mix the information content in packets before forwarding them. This mixing (or encoding) of information has been extensively studied over the last decade, e.g., [1], [2], [3], [4], [5]. While the advantages of network coding in the *multicast* setting are currently well understood, this is far from being the case when addressing the context of general network coding. Primarily, determining the capacity of a general network coding instance is a long standing open problem, e.g., [6], [7].

This work addresses the computational complexity of designing network codes that achieve or come close to achieving the network capacity. An instance to the `Network Coding` problem is a directed graph $G = (V, E)$, a set of source nodes $\{s_i\} \subseteq V$, a set of terminal nodes $\{t_j\} \subseteq V$, and a set of source/terminal requirements $\{(s_i, t_j)\}$ (implying that terminal t_j is interested in the information available at source s_i). In what follows we will consider acyclic graphs G, and follow standard definitions appearing for example in [8]. Each source s_i holds a message p_i that is to be transmitted to a certain subset of terminals. Each message is assumed to consist of k characters of a given finite alphabet Σ (also referred to as *packets*) and each edge of the network is assumed to have the capability of transmitting ℓ characters of Σ. We assume that each edge e is used at most once, namely at most ℓ packets are transmitted over e. With each edge $e = (u, v)$ we associate an encoding function g_e which ties the packets transmitted on edges entering u with the ℓ packets transmitted on e.

The objective is to define the encoding functions corresponding to edges in E such that terminal t_i will be able to decode the messages p_i it demands from the packets it receives on its incoming edges. More formally, we need to define for each terminal t_i a decoding function $\gamma_i : \Sigma^{d_i \ell} \to \Sigma^{r_i k}$ that enables t_i to decode the messages it demands from the information transmitted on its incoming edges (here d_i denotes the in-degree of t_i and r_i is the number of messages t_i requires). If such encoding and decoding functions exist, we say that the instance I to the `Network Coding` problem is (k, ℓ)-solvable over Σ. If the functions are linear we say that I has a *linear* or *vector linear* (k, ℓ)-solution over Σ. If the encoding and decoding functions are linear and $k = 1$ we say that the instance has a $(1, \ell)$-*scalar linear* solution over Σ.

Since, any (k, ℓ)-solution implies an $(rk, r\ell)$-solution for any integer r, we refer to the ratio $\frac{k}{\ell}$ as the rate of a (k, ℓ)-solution to I and denote the *capacity* $C(I)$ of I over Σ as the supremum of the ratio $\frac{k}{\ell}$ taken over (k, ℓ) solutions to I over Σ. Similarly, we define the linear capacity $C_l(I)$ and scalar linear capacity $C_{sl}(I)$ of I as the maximum rate achievable by vector-linear or scalar-linear solutions, respectively[1].

A. Previous work

Determining the capacity of a general `Network Coding` instance is a long standing open problem. Specifically, it is currently not known whether this problem is solvable in polynomial time, is NP-hard, or maybe it is even undecidable [9] (the undecidability assumes that the alphabet size can be arbitrary and unbounded). It is shown in [10] that determining the scalar linear capacity C_{sl} is an NP-hard problem. However, it is not known whether this holds for the vector-linear or

[1]Notice that $C(I)$, $C_l(I)$ and $C_{sl}(I)$ depend on Σ. To simplify our presentation, we omit Σ from our capacity notation.

general capacity, as the result of [10] does not extend to (k, ℓ) vector linear codes even for $k = 2$. In [8] it is shown that non-linear codes have an advantage over linear solutions as there exist instances in which linear codes do not suffice to achieve capacity. For specific instances to the `Network Coding` problem, it has been shown that combining combinatorial bounds and "Shannon-type" information inequalities suffice to characterize the capacity, e.g., [11], [12], although this is not the case in general [13].

B. *Our contribution*

In this work we initiate the study of *approximation* in the context of network coding. Namely, we show that given an instance I to the `Network Coding` problem of linear capacity $C_l(I)$, constructing a linear code with rate $\alpha C_l(I)$ for any universal constant α (independent of the size of the network) is "hard". Specifically, finding such network codes would solve a long standing open problem in the field of graph coloring. Our results apply to scalar linear codes and (k, ℓ) vector linear codes for constant values of k (independent of the network size). This implies the first hardness result for the general `Network Coding` problem that addresses vector linear solutions.

In addition, we consider the problem of determining the scalar linear capacity of a *planar* instance (an instance in which the underlying graph is planar). We show that even when the network is planar, this problem remains NP-hard. We note that the reduction presented in [10] does not result in a planar graph (and hence their results do not hold in the planar setting). We now state our main results in detail. We begin with some preliminaries on the topic of "graph coloring".

1) Graph coloring: An independent set in an undirected graph $G = (V, E)$ is a set of vertices that induce a subgraph which does not contain any edges. For an integer k, a k-coloring of G is a function $\sigma : V \to [1 \ldots k]$ which assigns colors to the vertices of G. A valid k-coloring of G is a coloring in which each color class is an independent set. The chromatic number $\chi(G)$ of G is the smallest k for which there exists a valid k-coloring of G. Finding $\chi(G)$ is a fundamental NP-hard problem. Hence, when limited to polynomial time algorithms, one turns to the question of estimating the value of $\chi(G)$ or to the closely related problem of *approximate coloring* in which one seeks to find a coloring of G with $r \cdot \chi(G)$ colors, for some approximation ratio $r \geq 1$, where the objective is to minimize r.

For a graph G of size n, the approximate coloring of G can be solved efficiently within an approximation ratio of $r = O\left(n \frac{(\log \log n)^2}{\log^3 n}\right)$ [14]. This result may seem rather weak — as a trivial approximation algorithm, with approximation ratio n, just colors each vertex with a different color. However, it turns out that one probably cannot do much better than the trivial algorithm. Namely, it is NP-hard to approximate $\chi(G)$ within a ratio of $n^{1-\varepsilon}$ for any constant $\varepsilon > 0$ [15]. Hence, there has been a long line of work addressing the coloring of graphs G which are known to have small chromatic number, e.g., [16], [17], [18], [19]. For example (and most relevant to our work), given a graph G which is known to be 3 colorable, the problem of coloring G with as few colors as possible has been extensively studied, e.g., [16], [17], [19].

The current state of affairs in the study of coloring 3 colorable graphs is an intriguing one. Results in this area have one of two flavors: "achievability" results, which specify an efficient algorithm for coloring the given graphs, or "lower bounds", which show that coloring these graphs with a small number of colors is a provably "hard" problem. On one hand, the currently best polynomial time algorithm [19] can color a 3 colorable graph G in roughly $n^{0.21}$ colors — a polynomial blowup! On the other hand, not much is known regarding lower bounds. It is NP-hard to color a 3-colorable graph G with 4 colors [20], [21]. Under stronger complexity assumptions (related to the *Unique Games Conjecture* [22]) it is hard to color a 3-colorable graph with any constant number of colors r [23] (here r is a universal constant independent of the size of the input graph). Resolving this gap between the upper and lower bounds presented above is a long standing open problem.

In this work we show that finding an approximate solution to the `Network Coding` problem of *constant* quality is at least as hard as coloring a 3 colorable graph with a constant number of colors, which in turn is at least as hard as the hardness assumptions specified in [23]. Regardless of the validity of the assumptions given in [23], our reduction shows that approximating the `Network Coding` problem within constant quality will solve a long standing open problem in approximate coloring.

2) Statement of results: Our main result can be summarized by the following theorem. In what follows, and throughout the paper, an *efficient* algorithm is one that runs in time polynomial in the instance size. Moreover, throughout, our alphabets Σ are assumed to be finite.

Theorem 1: Given any 3 colorable graph G, one can efficiently construct an instance I to the `Network Coding` problem with the following property. For any alphabet Σ, if one can efficiently find a linear (k, ℓ) solution to I over Σ that satisfies $\frac{k}{\ell} \geq \alpha C_l(I)$, then one can efficiently color G with $2^{3k/\alpha}$ colors (here $\alpha \leq 1$).

Corollary 2: Let $\alpha < 1$ be any constant. Let k be constant. Let Σ be any alphabet. If one can efficiently find a (k, ℓ) linear solution to every instance I of the `Network Coding` problem over Σ of rate $\alpha C_l(I)$, then one can efficiently color 3 colorable graphs with a constant number of colors.

A network coding instance is said to be *planar* if the underlying network can be drawn in the plane in such a way that no two edges cross each other. Planar graphs have seen a significant amount of research in the field of combinatorial optimization, e.g., [24], and many problems known to be "hard" on general graphs become significantly easier when the input graph is planar. One may suspect that the same occurs in the case of network coding. However, in this work we show:

Theorem 3: Given a planar instance I to the `Network Coding` problem, an alphabet Σ and a parameter ℓ, the problem of deciding whether $C_{sl}(I) \geq \frac{1}{\ell}$ is NP-complete.

The instances I implied by Theorems 1 and 3 are simple in nature and resemble the instances that have been used in the literature to express the advantage of network coding, e.g., [1], [25]. More specifically, given a graph G, the instances we construct correspond to the so-called `IndexCoding` problem recently studied by [26], [27]. The `IndexCoding` problem and its connection to network coding are described in Section II. We then turn to prove Theorem 1 and Theorem 3 in Sections III and IV respectively. Due to space limitations, some of our claims will appear without proof.

II. Preliminaries: The `IndexCoding` problem

The `IndexCoding` problem encapsulates the "source coding with side information" problem in which a single server wishes to communicate with several clients each having different side information. Formally, an instance to `IndexCoding` includes a set of clients $C = \{c_1, \ldots, c_n\}$ and a set of messages $P = \{p_1, p_2, \ldots, p_m\}$ to be transmitted by the server. Each client requires a certain subset of messages in P, while some messages in P are already available to it. Specifically, each client $c_i \in C$ is associated with two sets:

- $W(c_i) \subseteq P$ - the set of messages required by c_i.
- $H(c_i) \subseteq P$ - the set of messages available at c_i.

We refer to $W(c_i)$ and $H(c_i)$ as the "wants" and "has" sets of c_i, respectively. The server uses a broadcast channel to transmit messages to clients, each message is an encoding of messages in P. We assume that all messages transmitted by the server are received by all clients without an error. The objective is to design an encoding scheme which minimizes the number of transmissions.

We consider a *fractional* setting in which each message $p_i \in P$ consists of k packets $p_i^1, \ldots, p_i^k$ each a character of a given alphabet Σ. In each *round* of communication the server can transmit a single character of Σ (i.e., a single packet). The j'th round of communication is specified by an encoding function $g_j : \Sigma^{mk} \rightarrow \Sigma$. Namely, in the j'th round of communication the character $x_j = g_j(P)$ is transmitted by the server.

The goal in the `IndexCoding` problem is to find a set of encoding functions $\Phi = \{g_i\}_{i=1}^{\ell}$ that will allow each client to decode the messages it requested, while minimizing $\ell = |\Phi|$. More formally, we need to define for each client c_i a decoding function $\gamma_i : \Sigma^\ell \times (\Sigma^k)^{|H(c_i)|} \rightarrow (\Sigma^k)^{|W(c_i)|}$ that enables the client to decode the required messages in its "want" set from the transmitted messages and the messages in its "has" set. If such encoding and decoding functions exist, we say that the instance to `IndexCoding` is (k, ℓ)-solvable over Σ, and Φ is its solution. If the encoding and decoding functions are linear we say that the instance has a (k, ℓ)-*linear* solution over Σ. If in addition $k = 1$ we say that the instance has a $(1, \ell)$-*scalar linear* solution over Σ.

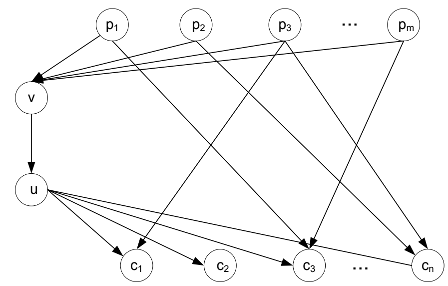

Fig. 1. The instance $\hat{I}$ of Proposition 4. There is a source node p_j for each message in P, a terminal node c_i for each client in C, and two additional nodes u and v. Each terminal c_i requires all messages from sources in $W(c_i)$. Each p_j is connected to v, u is connected to all terminals c_i, and v is connected to u. In addition we add edges (p_j, c_i) iff $p_j \in H(c_i)$. The broadcast channel corresponds to packets transmitted on (v, u), while the side information correspons to packets transmitted on edges (p_j, c_i).

As in the case of the `Network Coding` problem, any (k, ℓ)-solution implies an $(rk, r\ell)$-solution for any integer r. For an instance I to the `IndexCoding` problem, we refer to the ratio $\frac{k}{\ell}$ as the rate of a (k, ℓ)-solution to I and denote the capacity $\texttt{Opt}(I)$ of an instance I to `IndexCoding` over Σ as the supremum of the ratio $\frac{k}{\ell}$ taken over (k, ℓ) solutions to I over Σ. We also define by $\texttt{Opt}_l(I)$ and $\texttt{Opt}_{sl}(I)$ the vector-linear and scalar-linear capacities, i.e., capacities that can be achieved by using vector linear and scalar linear solutions, respectively.

There is a natural reduction from the `IndexCoding` problem to the problem of designing a network code for a certain network with general requirements. This connection can be summarized by the following Proposition.

Proposition 4: Let Σ be any alphabet. For every instance I to the `IndexCoding` problem one can efficiently construct an instance $\hat{I}$ to the `Network Coding` problem such that any (k, ℓ)-solution to $\hat{I}$ over Σ can be efficiently converted to a (k, ℓ)-solution to I and vice-versa. These conversions preserve linearity and thus $\texttt{Opt}_{sl}(I) = C_{sl}(\hat{I})$, $\texttt{Opt}_l(I) = C_l(\hat{I})$ and $\texttt{Opt}(I) = C(\hat{I})$ (over Σ).

To prove Proposition 4 we use the construction depicted in Figure 1 (due to space limitations full proof of the Proposition is omitted).

The `IndexCoding` problem was recently studied in [26], [27] where special instances of `IndexCoding` were considered. Namely, instances I in which $|P| = |C| = n$ and the only message client c_i wants is the message p_i ($W(c_i) = \{p_i\}$). In this case, the side information $\{H(c_i) \mid c_i \in C\}$ can be represented by a graph $G = (C, E)$ with vertex set C such that G contains an edge (i, j) if and only if client c_i

has message p_j (i.e., $p_j \in H(c_i)$). For instances I corresponding to undirected graphs G (in which edges are bi-directional) [26], [27] present certain connections between combinatorial properties of G and the value of $\texttt{Opt}(I)$. Specifically, [26], [27] show that: (a) A certain property of the adjacency matrix corresponding to G, referred to as the `Minrank`[2] [28] of G, characterizes the scalar linear capacity $\texttt{Opt}_{sl}(I)$. (b) This capacity is lower bounded by $1/\chi(G^c)$. Here G^c is the complement graph of G (i.e., an edge (i, j) appears in G iff it does not appear in G^c).

For an undirected graph G, and its corresponding instance to the `IndexCoding` problem, computing the `Minrank` of G (and thus $\texttt{Opt}_{sl}(I)$) was proven to be NP-complete in [29]. Namely (in our notation), using a reduction from the problem of 3-coloring, [29] shows that for any finite field Σ, determining whether $\texttt{Opt}_{sl}(I) \geq 1/3$ is NP-complete. Combining this result with Proposition 4 above, establishes that determining the scalar linear capacity of a general network coding instance is NP-hard. Notice that this may be viewed as an alternative proof to that given in [10]. We stress that the results in [29] do not imply inapproximability results of the nature presented in this work as in their reduction one can easily find a scalar linear solution which satisfies $\texttt{Opt}_{sl}(I) \geq 1/4$.

III. Proof of Theorem 1

In this section, we prove the following Theorem 5. By combining Theorem 5 with Proposition 4, we obtain Theorem 1 stated in the Introduction.

Theorem 5: Given any 3-colorable graph G, one can efficiently construct an instance I to the `IndexCoding` problem with the following property. For any alphabet Σ, if one can efficiently find a linear (k, ℓ)-solution to I over Σ that satisfies $\frac{k}{\ell} \geq \alpha\texttt{Opt}_l(I)$ then one can efficiently color G with $2^{3k/\alpha}$ colors (here $0 \leq \alpha \leq 1$).

Proof: Let $G = (V, E)$ be an undirected graph with $V = \{v_1, \ldots, v_n\}$. We start by defining a corresponding instance I to `IndexCoding` (as explained in Section II). The instance includes n clients $C = \{c_1, \ldots, c_n\}$ and n packets $P = \{p_1, \ldots, p_n\}$. For each client we define $W(c_i) = \{p_i\}$ and $H(c_i) = \{p_j | (i, j) \in E\}$, i.e., client c_i wants message i and has all messages wanted by its neighbors in G.

Assume that the complement graph G^c of G is 3-colorable (notice that the role of G in the theorem statement is played here by G^c). It follows that $\texttt{Opt}_{sl}(I) \geq \frac{1}{3}$ (for example by [29][3]). Assume also that we have an α approximation algorithm for constructing linear solutions to instances I of `IndexCoding`. Namely, that we can find a (k, ℓ) linear solution for I with $\frac{k}{\ell} \geq \alpha\texttt{Opt}_l(I) \geq \alpha\texttt{Opt}_{sl}(I) \geq \frac{\alpha}{3}$. This, in turn, implies that $\ell \leq 3k/\alpha$. Namely, for $j = 1, \ldots, \ell$ let $g_j \in \Sigma^{kn}$ characterize the solution to I in which the packet x_j transmitted in communication round j is equal to $\langle g_j, \bar{p}\rangle$. Here, $\langle\cdot\rangle$ is the standard inner product, and we denote by $\bar{p} \in \Sigma^{kn}$ the vector consisting of all kn packets in P.

We will now show, given the set $\{g_j\}_{j=1}^{\ell}$, how to construct a coloring of G^c of size $2^{\ell} \leq 2^{3k/\alpha}$. This will suffice to prove our assertion. Consider the first *generation* p_i^1 of each message in P (the first generation is the set that includes the first packet of every message in P). As the packets $\{x_j\}$ are a valid solution for instance I, it holds for each index $i = 1, \ldots, n$ that there is a vector $h_i \in span\{g_1, \ldots, g_\ell\}$ such that the vector $\langle h_i, \bar{p}\rangle$ may be used by client c_i in its decoding scheme when recovering p_i^1. Namely, the vector h_i has the following properties: (a) The coefficient in h_i corresponding to packet p_i^1 is non-zero. (b) For $p_j \notin (H(c_i) \cup W(c_i))$ the coefficients in h_i corresponding to all packets of message p_j are zero. We will use the vectors $\{h_1, \ldots, h_n\}$ to find a coloring of G^c.

We say that two vectors h_{i_1} and h_{i_2} are *component equivalent* if for every j the j'th entry of h_{i_1} differs from 0 iff the same holds for h_{i_2}. Consider two indices i_1 and i_2 for which h_{i_1} and h_{i_2} are component equivalent. We claim that it must be the case that the corresponding vertices v_{i_1} and v_{i_2} share an edge in G. This follows since the coefficient in h_{i_1} corresponding to $p_{i_1}^1$ is non-zero. This, in turn, implies that it must be the case that $i_1 \in H(c_{i_2})$. The latter implies that $(v_{i_1}, v_{i_2}) \in E$.

We conclude that for each $h \in span\{g_1, \ldots, g_\ell\}$, the vertices v_i with corresponding vectors h_i that are component equivalent to h form a clique in G. This implies that G has a clique cover[4] of size $2^{\ell} \leq 2^{3k/\alpha}$, or alternatively that G^c can be colored with $2^{3k/\alpha}$ colors. It is not hard to verify that both the vectors $\{h_1, \ldots, h_n\}$ and the coloring of G^c can be efficiently deduced from the solution $\{g_j\}_{j=1}^{\ell}$ to I. This concludes our proof. ■

IV. Proof of Theorem 3

We now turn to prove Theorem 3. Our starting point is a reduction presented in [29] between a given undirected graph G and an instance I to the `IndexCoding` problem[5] that satisfies $\chi(G) \leq 3$ iff $\texttt{Opt}_{sl}(I) \geq 1/3$. In what follows, we present a reduction between the instance I and a planar instance I_p to the `Network Coding` problem which satisfies $C_{sl}(I_p) \geq 1$ if and only if $\texttt{Opt}_{sl}(I) \geq 1/3$. This will suffice to prove Theorem 3.

The instance I of [29] includes a set of n clients $C = \{c_1, \ldots, c_n\}$ and n messages $P = \{p_1, \ldots, p_n\}$. Moreover, each client c_i only wants the single message p_i, i.e., $W(c_i) = \{p_i\}$.

[2] Due to space limitations we will not define and discuss the notion of `Minrank`.

[3] As G^c is 3-colorable, G can be covered by 3 cliques. For each such clique C, define the corresponding transmitted packet $x_C = \sum_{i \in C} p_i$.

[4] A clique cover of a graph G is a collection of subsets of vertices $U_1, \ldots, U_k$ such that $\cup_i U_i = V$ and the subgraph of G induced by each U_i is a clique.

[5] To be more precise, the reduction presented in [29] is between the problem of graph coloring and the `Minrank` problem; the extension of the reduction to `IndexCoding` is straightforward.

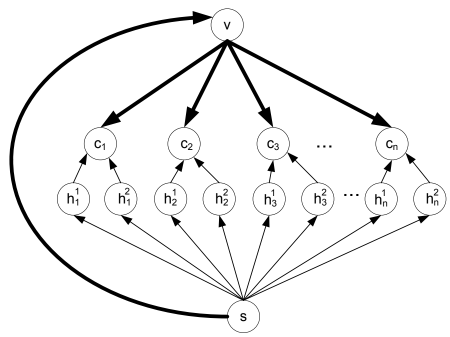

Fig. 2. A reduction from Problem `IndexCoding` to Problem `Network Coding`. The bold edges denote three parallel edges of unit capacity.

We proceed to construct an instance I_p to the `Network Coding` problem. We begin by describing the structure of the underlying planar network $G_p(V, E)$ (schematically depicted in Figure 2). The set of nodes V of G_p includes a node c_i for each client $c_i \in C$, and two additional vertices, v and s. In addition, for each client $c_i \in C$ we add $|H(c_i)|$ vertices $h_i^1, \dots, h_i^{|H(c_i)|}$ that correspond to elements of $H(c_i)$. The set of edges E of G_p is constructed as follows. First, we connect s to v by three parallel edges. Next, v is connected to each node c_i by three parallel edges. For each client $c_i \in C$ we connect each vertex in $\{h_i^1, \dots, h_i^{|H(c_i)|}\}$ to vertex c_i by a single edge. Finally, the node s is connected to the nodes that correspond to $\{H(c_i) \; |c_i \in C\}$.

The instance includes a set of n sources $\{s_i\}_{i=1}^n$ that correspond to messages in P (source s_i having message p_i). All sources are co-located at vertex s. For each client $c_i \in C$ we add $1 + |H(c_i)|$ terminals such that one terminal is located at node c_i and requires packet p_i from source s_i (this corresponds to the message p_i wanted by c_i in I). The rest of the terminals are located at vertices $\{h_i^1, \dots, h_i^{|H(c_i)|}\}$. Terminal h_i^j will require the j'th message in the set $H(c_i)$ (from the corresponding source). The latter terminals will force the sources to transmit the "side information" of I on the edges (s, h_i^j).

Clearly, $G_p(V, E)$ is planar and any (scalar linear) (1,3)-code for I corresponds to a (scalar linear) (1,1)-network code for I_p. Also, it is not hard to verify that a (scalar linear) (1,1)-network code for I_p corresponds to a (scalar linear) (1,3)-solution for I, which completes the proof of the theorem.

References

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung. Network Information Flow. *IEEE Transactions on Information Theory*, 46(4):1204–1216, 2000.

[2] S.-Y. R. Li, R. W. Yeung, and N. Cai. Linear Network Coding. *IEEE Transactions on Information Theory*, 49(2):371 – 381, 2003.

[3] R. Koetter and M. Medard. An Algebraic Approach to Network Coding. *IEEE/ACM Transactions on Networking*, 11(5):782 – 795, 2003.

[4] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen. Polynomial Time Algorithms for Multicast Network Code Construction. *IEEE Transactions on Information Theory*, 51(6):1973–1982, June 2005.

[5] T. Ho, M. Médard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong. A Random Linear Network Coding Approach to Multicast. *IEEE Transactions on Information Theory*, 52(10):4413–4430, 2006.

[6] T. Cover and J. Thomas. *Elements of Information Theory, second edition*. John Wiley and Sons, 2006.

[7] R. W. Yeung, S.Y.R. Li, N. Cai, and Z. Zhang. *Network Coding Theory*. Now Publishers Inc, 2006.

[8] R. Dougherty, C. Freiling, and K. Zeger. Insufficiency of Linear Coding in Network Information Flow. *IEEE Transactions on Information Theory*, 51(8):2745–2759, 2005.

[9] A. Rasala Lehman. *Network Coding*. Ph.d. thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 2005.

[10] A. Lehman and E. Lehman. Complexity Classification of Network Information Flow Problems. In *Proceedings of SODA*, 2004.

[11] M. Adler, N. J. A. Harvey, K. Jain, R. D. Kleinberg, and A. L. Rasala. On the Capacity of Information Networks. *In Proc. ACM-SIAM Symp. Discrete Algorithms (SODA)*, pages 2345–2364, 2006.

[12] N. J. A. Harvey, R. D. Kleinberg, , and A. Rasala Lehman. On the Capacity of Information Networks. *IEEE Transactions on Information Theory*, 52(6):23452364, 2006.

[13] R. Dougherty, C. Freiling, and K. Zeger. Networks, Matroids, and Non-Shannon Information Inequalities. *IEEE Transactions on Information Theory*, 53(6):1949–1969, 2007.

[14] M. Halldorsson. A Still Better Performance Guarantee for Approximate Graph Coloring. *Information Processing Letters*, 45:19–23, 1993.

[15] D. Zuckerman. Linear Degree Extractors and the Inapproximability of Max Clique and Chromatic Number. *In Proceedings of the 38th annual ACM symposium on Theory of Computing*, pages 681–690, 2006.

[16] D. Karger, R. Motwani, and M. Sudan. Approximate Graph Coloring by Semidefinite Programming. *Journal of ACM*, 45(2):246–265, 1998.

[17] A. Blum and D. Karger. An $\tilde{O}(n^{3/14})$-coloring for 3-colorable Graphs. *Information Processing Letters*, 61(6):49–53, 1997.

[18] E. Halperin, R. Nathaniel, and U. Zwick. Coloring k-colorable graphs using smaller palettes. *Journal of Algorithms*, 45:72–90, 2002.

[19] S. Arora, E. Chlamtac, and M. Charikar. New Approximation Guarentee for Chromatic Number. *In proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 215–224, 2006.

[20] S. Khanna, N. Linial, and S. Safra. On the Hardness of Approximating the Chromatic Number. *Combinatorica*, 20:393–415, 2000.

[21] V. Guruswami and S. Khanna. On the Hardness of 4-coloring a 3-colorable Graph. *Proceedings of the 15th Annual IEEE Conference on Computational Complexity*, pages 188–197, 2000.

[22] S. Khot. On the Power of Unique 2-prover 1-round Games. In *Proceedings of the 34th annual ACM symposium on Theory of computing*, pages 767–775, 2002.

[23] I. Dinur, E. Mossel, and O. Regev. Conditional Hardness for Approximate Coloring. In *Proceedings of 38th ACM Symposium on Theory of Computing*, pages 344–353, 2006.

[24] B. S. Baker. Approximation Algorithms for NP-complete Problems on Planar Graphs. *J. ACM*, 41(1):153–180, 1994.

[25] Z. Li and B. Li. Network Coding: The Case for Multiple Unicast Sessions, 2004.

[26] Z. Bar-Yossef, Y. Birk, T. S. Jayram, and T. Kol. Index Coding with Side Information. *In Proceedings of 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 197–206, 2006.

[27] E. Lubetzky and U. Stav. Non-linear Index Coding Outperforming the Linear Optimum. *In Proceedings of 48th Annual IEEE Symposium on Foundations of Computer Science*, pages 161–168, 2007.

[28] W. H. Haemersl. On Some Problems of Lovász Concerning the Shannon Capacity of a Graph. *IEEE Transactions on Information Theory,*, 25(2):231–232, 1979.

[29] R. Peeters. Orthogonal Representations Over Finite Fields and the Chromatic Number of Graphs. *Combinatorica*, 16(3):417–431, 1996.