Improved Approximation Algorithms for Index Coding

Dror Chawin*   Ishay Haviv*

Abstract

The index coding problem is concerned with broadcasting encoded information to a collection of receivers in a way that enables each receiver to discover its required data based on its side information, which comprises the data required by some of the others. Given the side information map, represented by a graph in the symmetric case and by a digraph otherwise, the goal is to devise a coding scheme of minimum broadcast length.

We present a general method for developing efficient algorithms for approximating the index coding rate for prescribed families of instances. As applications, we obtain polynomial-time algorithms that approximate the index coding rate of graphs and digraphs on n vertices to within factors of $O(n/\log^2 n)$ and $O(n/\log n)$ respectively. This improves on the approximation factors of $O(n/\log n)$ for graphs and $O(n \cdot \log\log n/\log n)$ for digraphs achieved by Blasiak, Kleinberg, and Lubetzky (IEEE Trans. Inform. Theory, 2013). For the family of quasi-line graphs, we exhibit a polynomial-time algorithm that approximates the index coding rate to within a factor of 2. This improves on the approximation factor of $O(n^{2/3})$ achieved by Arbabjolfaei and Kim (ISIT, 2016) for graphs on n vertices taken from certain sub-families of quasi-line graphs.

Our approach is applicable for approximating a variety of additional graph and digraph quantities to within the same approximation factors. Specifically, it captures every graph quantity sandwiched between the independence number and the clique cover number and every digraph quantity sandwiched between the maximum size of an acyclic induced sub-digraph and the directed clique cover number.

1   Introduction

The index coding problem, introduced in 1998 by Birk and Kol [8], is a central problem in network information theory, motivated by various applications ranging from satellite and wireless communications to network coding and distributed storage (see, e.g., [4]). The problem involves a sender that holds an n-symbol message $x \in \Sigma^n$ over some alphabet Σ and n receivers $R_1, \ldots, R_n$, where the ith receiver R_i is interested in the ith symbol x_i. Every receiver R_i is given as side information the restriction $x_{N(i)}$ of x to the indices that lie in some set $N(i) \subseteq [n] \setminus \{i\}$. The side information map is represented by a digraph G on the vertex set $[n]$ that includes a directed edge (i, j) if $j \in N(i)$, that is, if the receiver R_i knows x_j. The sender wishes to broadcast to the receivers a short encoded message that enables each receiver R_i to discover x_i based on the transmitted message and on the given side information.

An index code for a digraph G on the vertex set $[n]$ over an alphabet Σ is formally defined as an encoding function $E : \Sigma^n \to \Sigma_E$ for some alphabet Σ_E, such that for each $i \in [n]$, there exists a decoding function $g_i : \Sigma_E \times \Sigma^{|N(i)|} \to \Sigma$ satisfying $g_i(E(x), x_{N(i)}) = x_i$ for all $x \in \Sigma^n$. Note that the encoding length in bits of such an index code is $\lceil \log_2 |\Sigma_E| \rceil$. For a side information digraph G and for an integer $t \geq 1$, let $\beta_t(G)$ denote the smallest possible encoding length in bits of an index code for G over an alphabet Σ of size $|\Sigma| = 2^t$. The index coding rate of G, denoted by $\beta(G)$, expresses the average asymptotic number of encoding bits needed per bit in each input block for the side information digraph G. It is therefore defined by $\beta(G) = \lim_{t \to \infty} \frac{\beta_t(G)}{t}$, where the existence of the limit follows by combining the sub-additivity of β_t with Fekete's lemma (see, e.g., [25, Lemma 11.6]). The quantities β_t and β are naturally extended to (undirected) graphs by replacing every undirected edge by two oppositely directed edges.

A simple upper bound on the index coding rate of a digraph is its (directed) clique cover number. Here, a clique of a digraph $G = (V, E)$ is a set $C \subseteq V$ such that every two distinct vertices of C are connected by two oppositely directed edges. For a digraph G and a clique cover $\mathcal{C}$ of G, consider the encoding function over $\Sigma = \{0, 1\}$ that for every clique $C \in \mathcal{C}$ includes the xor of the bits associated with C. This encoding function forms an index code for G, because each receiver is able to discover its bit given its side information and the xor associated with a clique that covers its vertex. We refer to such an index code as a clique cover index code. It follows that every digraph G satisfies $\beta(G) \leq \beta_1(G) \leq \mathrm{CC}(G)$, where $\mathrm{CC}(G)$ stands for the minimum size of a clique cover of G. This upper bound on the index coding rate has multiple stronger variants, e.g., a local relaxation and a fractional relaxation based on linear programming (see, e.g., [4, Section 6]). On the other hand, the index coding rate $\beta(G)$ of a digraph G is known to satisfy $\beta(G) \geq \mathrm{MAIS}(G)$, where $\mathrm{MAIS}(G)$ stands for the maximum size of an acyclic induced sub-digraph of G (see [6, 9]). Note that in the undirected setting, the above bounds can be written as $\alpha(G) \leq \beta(G) \leq \overline{\chi}(G)$, where $\alpha(G)$ and $\overline{\chi}(G)$ denote, respectively, the independence and clique cover numbers of a graph G.

From a computational point of view, the difficulty of determining the value of $\beta(G)$ for a given graph or digraph G has motivated researchers to suggest efficient approximation algorithms for the problem. A result of Blasiak, Kleinberg, and Lubetzky [9] implies that there exists a polynomial-time algorithm that given a digraph G on n vertices approximates the value of $\beta(G)$ to within a factor of $O(n \cdot \frac{\log \log n}{\log n})$, improving on the trivial factor of n. It was further noted in [9, Remark V.2] that for graphs on n vertices, it is possible to achieve in polynomial time a better approximation factor of $O(\frac{n}{\log n})$. The proofs in [9] further imply that the fractional clique cover number of a digraph on n vertices approximates the index coding rate to within a factor of $O(n \cdot \frac{\log \log n}{\log n})$, and that the clique cover number of a graph on n vertices approximates the index coding rate to within a factor of $O(\frac{n}{\log n})$ (see also [4, Propositions 9.1 and 9.3]). The challenge of improving these approximation factors in the directed and undirected settings was explicitly raised in [4, Open Problem 9.1]. By combining an approach of [9] with results from Ramsey theory [7], Arbabjolfaei and Kim [3] showed that for certain families of graphs, the clique cover number approximates the index coding rate to within tighter factors. In particular, they obtained a factor of $O(n^{2/3})$ for graphs on n vertices from the families of line graphs and fuzzy circular interval graphs (see, e.g., [13]).

1.1 Our Contribution

In this paper, we offer a general method for developing efficient algorithms for approximating the index coding rate for prescribed families of instances. To do so, we consider the algorithmic problem, which might be of independent interest, of finding economical clique covers of digraphs, where the approximation guarantee is measured with respect to the maximum size of an acyclic induced sub-digraph of the digraph at hand. This is in contrast to the standard setting, where the size of the produced clique cover is measured with respect to the digraph's clique cover number. Our results on this problem yield efficient approximation algorithms for a variety of graph and digraph quantities. In what follows, we present those results and describe their consequences for the index coding problem.

1.1.1 Clique cover vs. Acyclic induced sub-digraph

A family of graphs or digraphs is called hereditary if it is closed under removal of vertices (but not necessarily under removal of edges). Consider the problem that given a digraph G on n vertices, taken from a prescribed hereditary digraph family $\mathcal{G}$, asks to find a clique cover of G whose size does not exceed $\gamma \cdot \mathrm{MAIS}(G)$ for as small as possible $\gamma = \gamma(n)$. In the present paper, we exhibit a general method for tackling this problem, from both the existential and algorithmic perspectives, where the guaranteed factor γ depends on the Ramsey numbers of the family $\mathcal{G}$ (see Definition 1.2). Note that under this framework, one can also consider families of (undirected) graphs, represented as digraphs in which adjacent vertices are connected by two oppositely directed edges. In this representation, the maximum size of an acyclic induced sub-digraph is simply the independence number of the corresponding graph.

Before describing our general method, let us mention a well-understood special case of the above problem, where $\mathcal{G}$ is the family of all graphs. Erdős [15] proved in 1967 that for every graph G on n vertices, it holds that $\overline{\chi}(G) \leq O(\frac{n}{\log^2 n}) \cdot \alpha(G)$. The bound is optimal up to a multiplicative constant because, as is well known, a random graph G on n vertices satisfies $\alpha(G) = \Theta(\log n)$ and $\overline{\chi}(G) = \Theta(\frac{n}{\log n})$ (see, e.g., [2, Chapter 10]). In the early nineties, Boppana and Halldórsson [10] proved the following algorithmic form of Erdős's result.

Theorem 1.1 ([10, Theorem 3]). *There exists a polynomial-time algorithm that given a graph G on n vertices finds a clique cover of G of size $O(\frac{n}{\log^2 n}) \cdot \alpha(G)$.*

To state our general method, we need the following definition of Ramsey numbers, which extends a definition of Belmonte, Heggernes, van 't Hof, Rafiey, and Saei [7] to the directed setting.

Definition 1.2 (Ramsey Numbers). *For a graph family $\mathcal{G}$ and for two integers $s, t \geq 1$, let $R_{\mathcal{G}}(s,t)$ denote the smallest integer r such that every member of $\mathcal{G}$ on at least r vertices has a clique of size s or an independent set of size t. Similarly, for a digraph family $\mathcal{G}$ and for two integers $s, t \geq 1$, let $\vec{R}_{\mathcal{G}}(s,t)$ denote the smallest integer r such that every member of $\mathcal{G}$ on at least r vertices has a clique of size s or an acyclic induced sub-digraph of size t.*

Throughout the paper, for a digraph G and for a set of vertices I, we let $G[I]$ denote the sub-digraph of G induced by I. The same notation is used for graphs.

Our method is described by the following theorem, which generalizes the approach of [10].

Theorem 1.3. *Let $\mathcal{G}$ be a hereditary family of digraphs, and let $Q : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be a function such that $\vec{R}_{\mathcal{G}}(s,t) \leq Q(s,t)$ for all integers $s,t \geq 1$. Let $f_Q : \mathbb{N} \to \mathbb{N}$ denote the function defined by*

$$f_Q(n) = \min_{s,t \in \mathbb{N}} \{s \cdot t \mid Q(s+1, t+1) > n\}.$$

Then for every digraph $G \in \mathcal{G}$ on n vertices, it holds that

$$\mathrm{CC}(G) \leq \Big(\sum_{i=1}^{n} \frac{1}{f_Q(i)} \Big) \cdot \mathrm{MAIS}(G). \tag{1}$$

Suppose further that there exists a polynomial-time algorithm that given a digraph $G \in \mathcal{G}$ on n vertices finds a clique C of G and a set I for which $G[I]$ is acyclic, such that for all integers $s, t \geq 1$, if $n \geq Q(s,t)$ then either $|C| \geq s$ or $|I| \geq t$. Then there exists a polynomial-time algorithm that given a digraph $G \in \mathcal{G}$ on n vertices finds a clique cover of G whose size does not exceed the bound in (1).

Note that if one is interested only in the existential statement of the theorem, given in (1), then the function Q can be taken to be equal to $\vec{R}_{\mathcal{G}}$. However, to obtain the algorithmic statement of the theorem, it might be needed to consider some function Q with larger values.

We demonstrate the usefulness of Theorem 1.3 by applying it to several families of graphs and digraphs. Let us first mention that the aforementioned results of [15] and [10] (Theorem 1.1) can be derived by applying Theorem 1.3 to the family of all graphs, using a classical argument from Ramsey theory due to Erdős and Szekeres [17]. For an analogue result for digraphs, we combine Theorem 1.3 with a directed variant of the argument of [17] and prove the following.

Theorem 1.4. *There exists a polynomial-time algorithm that given a digraph G on n vertices finds a clique cover of G of size $O(\frac{n}{\log n}) \cdot \mathrm{MAIS}(G)$.*

The guarantee of the algorithm given by Theorem 1.4 is optimal up to a multiplicative constant, because there exist digraphs on n vertices whose ratio between the clique cover number and the maximum size of an acyclic induced sub-digraph is $\Theta(\frac{n}{\log n})$. This indeed follows from a result of Erdős and Moser [16], who proved that there exist tournaments G on n vertices, i.e., digraphs with exactly one edge connecting every pair of vertices, satisfying $\mathrm{MAIS}(G) \leq 2 \cdot \lfloor \log_2 n \rfloor + 1$. Since a tournament has no clique of size larger than 1, those digraphs G satisfy $\mathrm{CC}(G) = n$, and thus attain the claimed ratio.

We further apply Theorem 1.3 to hereditary digraph families $\mathcal{G}$ whose Ramsey numbers $\vec{R}_{\mathcal{G}}(s,t)$ grow polynomially with s and t (see Theorem 2.3). As an application, we improve the upper bounds of [3] on the ratio between the clique cover and independence numbers for hereditary graph families with polynomially bounded Ramsey numbers (see Corollary 2.5 and the discussion that follows it). In particular, for hereditary graph families $\mathcal{G}$ satisfying $R_{\mathcal{G}}(s,t) = O(s \cdot t)$, we show that every graph $G \in \mathcal{G}$ on n vertices satisfies $\overline{\chi}(G) \leq O(\log n) \cdot \alpha(G)$. This particular setting is motivated by a paper of Belmonte et al. [7], who proved that $R_{\mathcal{G}}(s,t) = O(s \cdot t)$ for $\mathcal{G}$ being either the family of line graphs or the family of fuzzy circular interval graphs (see, e.g., [13]).

In fact, for these two aforementioned graph families we improve on the logarithmic multiplicative term deduced from Theorem 1.3 and obtain a much stronger bound along with an algorithmic result. To do so, we consider the family of quasi-line graphs, those graphs where the neighborhood of every vertex can be partitioned into two cliques. This family of graphs, which plays a central

role in the structural characterization of claw-free graphs due to Chudnovsky and Seymour [13], contains the family of line graphs as well as the family of fuzzy circular interval graphs. We prove the following theorem.

Theorem 1.5. *There exists a polynomial-time algorithm that given a quasi-line graph G finds a clique cover of G of size at most $2 \cdot \alpha(G)$.*

The guarantee of the algorithm given by Theorem 1.5 is essentially optimal, even for line graphs, because the ratio between the clique cover and independence numbers of such graphs can be arbitrarily close to 2 when the number of vertices grows (see Remark 3.1). Since every graph G satisfies $\alpha(G) \leq \overline{\chi}(G)$, Theorem 1.5 implies that it is possible to approximate the clique cover number of quasi-line graphs to within a factor of 2 in polynomial time. This extends a recent result on line graphs due to Daneshpajouh, Meunier, and Mizrahi [14].

1.1.2 Approximation algorithms for Index Coding

The previous section offers efficient algorithms that given a digraph G on n vertices, taken from some specified digraph family, find a clique cover of G of size at most $\gamma \cdot \mathrm{MAIS}(G)$ for some $\gamma = \gamma(n)$. Consequently, these algorithms can be used to approximate to within a factor of γ any digraph quantity ψ that satisfies $\mathrm{MAIS}(G) \leq \psi(G) \leq \mathrm{CC}(G)$ for every digraph G. In the undirected setting, those algorithms can be used to approximate any graph quantity ψ that satisfies $\alpha(G) \leq \psi(G) \leq \overline{\chi}(G)$ for every graph G. It therefore follows that the algorithms from the previous section can be used to efficiently approximate the index coding rate β of a given graph or digraph. Moreover, the same approximation guarantees are achievable for various additional index coding measures, e.g., the scalar capacity β_1, the $\beta^\star$ capacity due to Alon et al. [1], the minrank parameter over a field $\mathbb{F}$ due to Haemers [19] (which characterizes the optimal length of linear index codes over $\mathbb{F}$ [6]), and many more. The approximation also applies to graph and digraph quantities with the above sandwich properties that are not directly related to index coding, such as the Shannon capacity of graphs [24] and the Sperner capacity of (complement) digraphs [21].

For concreteness, we state below our results on approximating the index coding rate β. We start with the case of general graphs. As a consequence of the result of Boppana and Halldórsson [10], stated earlier as Theorem 1.1, we obtain the following.

Theorem 1.6. *There exists a polynomial-time algorithm that given a graph G on n vertices finds a clique cover index code for G of length $O(\frac{n}{\log^2 n}) \cdot \beta(G)$, and in particular, approximates the value of $\beta(G)$ to within a factor of $O(\frac{n}{\log^2 n})$.*

Note that Theorem 1.6 improves on the approximation factor reported in [9, Remark V.2] and in [4, Propositions 9.1] by a multiplicative term of $\log n$.

We proceed with the case of general digraphs. As a consequence of Theorem 1.4, we obtain the following.

Theorem 1.7. *There exists a polynomial-time algorithm that given a digraph G on n vertices finds a clique cover index code for G of length $O(\frac{n}{\log n}) \cdot \beta(G)$, and in particular, approximates the value of $\beta(G)$ to within a factor of $O(\frac{n}{\log n})$.*

The algorithm given by Theorem 1.7 improves on the algorithm of Blasiak et al. [9] for digraphs in two respects. Firstly, the approximation factor is improved by a multiplicative term of $\log \log n$. Secondly, our algorithm produces an index code that corresponds to a clique cover of the input digraph rather than to a fractional clique cover. Therefore, the algorithm applies to additional index coding measures such as the scalar capacity β_1 and the minrank over a field $\mathbb{F}$, which are not bounded from above by the fractional clique cover. On the other hand, the algorithm of [9] enjoys the advantage that it extends to a generalized version of the index coding problem, which allows the number of receivers m to be larger than the number of symbols n (see Section 4). We extend Theorem 1.7 to this generalized setting, however, this extension does not preserve the approximation factor. Yet, it produces index codes whose rate is smaller than the rate guaranteed by the algorithm of [9] whenever the number of receivers m satisfies $m = o(n \cdot \log \log n)$. For the precise statement, see Theorem 4.1.

Finally, we consider the restriction of the index coding problem to the family of quasi-line graphs. As a consequence of Theorem 1.5, we obtain the following.

Theorem 1.8. *There exists a polynomial-time algorithm that given a quasi-line graph G finds a clique cover index code for G of length at most $2 \cdot \beta(G)$, and in particular, approximates the value of $\beta(G)$ to within a factor of 2.*

As mentioned earlier, the family of quasi-line graphs contains the family of line graphs and the family of fuzzy circular interval graphs. Therefore, Theorem 1.8 improves, in the existential and algorithmic manners, the approximation factor of $O(n^{2/3})$ achieved in [3] for graphs on n vertices from these families.

It would be intriguing to determine whether the approximation factors guaranteed by Theorems 1.6, 1.7, and 1.8 can be improved. A viable strategy for addressing this challenge would be to employ stronger lower and upper bounds on the index coding rate than the ones utilized here (namely, the maximum size of an acyclic induced sub-digraph and the clique cover number).

1.2 Related Work

We gather here several algorithmic and hardness results from the literature regarding graph quantities considered in the present paper. The independence and clique cover numbers of graphs on n vertices can be approximated to within a factor of $O(n \cdot \frac{(\log \log n)^2}{\log^3 n})$ in polynomial time [18, 20], and it is NP-hard to approximate them to within a factor of $n^{1-\varepsilon}$ for any $\varepsilon > 0$ [26].

For the index coding problem, it was shown in [22] that assuming a variant of the Unique Games Conjecture, it is hard to approximate the value of $\beta_t(G)$ for an input graph G to within any constant factor for any fixed integer $t \geq 1$. It was further shown in [11] that it is NP-hard to approximate the optimal length of a linear index code over any fixed finite field to within any constant factor (see also [12]).

It is worth noting that it is NP-hard to approximate the minimum size of a vertex cover of triangle-free graphs to within a factor of 1.36 [5]. As observed in [14], hardness for the vertex cover problem on triangle-free graphs implies hardness for the clique cover problem on line graphs. To see this, recall that the line graph of a graph G is the graph whose vertices are the edges of G, with an edge between every two edges of G that share a common vertex. It can be verified that if G

is triangle-free, then the minimum size of a vertex cover of G is equal to the minimum size of a clique cover of its line graph. The result of [5] therefore implies that it is NP-hard to approximate the clique cover number of line graphs (and thus of quasi-line graphs) to within a factor of 1.36.

1.3 Outline

The rest of the paper is organized as follows. In Section 2, we present our general method for finding clique covers of digraphs and prove Theorem 1.3. We then apply the theorem to several families of digraphs, and in particular, prove Theorem 1.7. We also present the method for graphs and show that it generalizes Theorem 1.1 of [10] (with the full details given in Appendix A). In Section 3, we present our algorithm for finding clique covers of quasi-line graphs and prove Theorem 1.5. Finally, in Section 4, we use Theorem 1.7 to offer an algorithm for a generalized version of the index coding problem.

2 Clique Covering of Digraphs

In this section, we present our general method for finding clique covers of digraphs, confirm Theorem 1.3, and demonstrate its applicability. We start with the following lemma, which produces clique covers of digraphs from a given hereditary family and analyzes their size through a standard argument (see, e.g., [20, Lemma 2]).

Lemma 2.1. *Let $\mathcal{G}$ be a hereditary family of digraphs, and let $f : \mathbb{N} \to \mathbb{R}^+$ be a monotone non-decreasing function. Suppose that every digraph $G \in \mathcal{G}$ on n vertices has a clique of size at least $\frac{f(n)}{\mathrm{MAIS}(G)}$. Then for every digraph $G \in \mathcal{G}$ on n vertices, it holds that*

$$\mathrm{CC}(G) \leq \Big(\sum_{i=1}^{n} \frac{1}{f(i)} \Big) \cdot \mathrm{MAIS}(G).　\qquad (2)$$

Moreover, if there exists a polynomial-time algorithm that given a digraph $G \in \mathcal{G}$ on n vertices finds a clique of G of size at least $\frac{f(n)}{\mathrm{MAIS}(G)}$, then there exists a polynomial-time algorithm that given a digraph $G \in \mathcal{G}$ on n vertices finds a clique cover of G whose size does not exceed the bound in (2).

Proof: Let $\mathcal{G}$ be a hereditary family of digraphs, and let $f : \mathbb{N} \to \mathbb{R}^+$ be a function as in the lemma. Consider some digraph $G \in \mathcal{G}$ on n vertices. We define a clique cover of G by repeatedly choosing a clique of G and removing its vertices.

More specifically, we define a sequence of induced sub-digraphs $G_1, \ldots, G_r$ of G and a sequence of cliques $C_1, \ldots, C_r$ of G as follows. Set $G_1 = G$. For each $j \geq 1$, if G_j has at least one vertex, let n_j denote the number of its vertices, and use the fact that $G_j \in \mathcal{G}$ to obtain that there exists a clique C_j of G_j of size

$$|C_j| \geq \frac{f(n_j)}{\mathrm{MAIS}(G_j)} \geq \frac{f(n_j)}{\mathrm{MAIS}(G)},　\qquad (3)$$

where the second inequality holds because G_j is an induced sub-digraph of G. We proceed to the next iteration of the process with the digraph G_{j+1} obtained from G_j by removing the vertices of C_j. Since $\mathcal{G}$ is hereditary, it holds that $G_{j+1} \in \mathcal{G}$. Note that G_{j+1} has $n_j - |C_j|$ vertices.

Letting r denote the number of cliques produced until no vertices remain, the sets $C_1, \ldots, C_r$ form a clique cover of G of size r, hence $\mathrm{CC}(G) \leq r$. Combining (3) with the assumption that the function f is monotone non-decreasing, we obtain that for each $j \in [r]$, it holds that

$$1 = |C_j| \cdot \frac{1}{|C_j|} \leq |C_j| \cdot \frac{1}{f(n_j)} \cdot \mathrm{MAIS}(G) \leq \left(\sum_{i=0}^{|C_j|-1} \frac{1}{f(n_j - i)} \right) \cdot \mathrm{MAIS}(G).$$

By summing the above over all $j \in [r]$, we obtain that

$$\mathrm{CC}(G) \leq r \leq \left(\sum_{j=1}^{r} \sum_{i=0}^{|C_j|-1} \frac{1}{f(n_j - i)} \right) \cdot \mathrm{MAIS}(G) = \left(\sum_{i=1}^{n} \frac{1}{f(i)} \right) \cdot \mathrm{MAIS}(G),$$

where the equality holds because $n_1 = n$, $n_{j+1} = n_j - |C_j|$ for all $j \in [r-2]$, and $n_r = |C_r|$.

Finally, observe that the iterative process described above, whose number of iterations does not exceed the number of vertices of G, confirms the algorithmic statement of the lemma and completes the proof. ∎

Equipped with Lemma 2.1, we are ready to prove Theorem 1.3 (recall Definition 1.2).

Proof of Theorem 1.3: Let $\mathcal{G}$ be a hereditary family of digraphs, and let $Q : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be a function such that $\vec{R}_\mathcal{G}(s,t) \leq Q(s,t)$ for all integers $s, t \geq 1$. Recall that $f_Q : \mathbb{N} \to \mathbb{N}$ is the function defined by

$$f_Q(n) = \min_{s,t \in \mathbb{N}} \{ s \cdot t \mid Q(s+1, t+1) > n \},$$

and notice that it is monotone non-decreasing.

We first claim that every digraph $G \in \mathcal{G}$ on n vertices has a clique of size at least $\frac{f_Q(n)}{\mathrm{MAIS}(G)}$. To see this, denote by $\omega(G)$ the maximum size of a clique of G, and let us observe that

$$Q(\omega(G) + 1, \mathrm{MAIS}(G) + 1) > n. \tag{4}$$

Indeed, otherwise it would follow that

$$\vec{R}_\mathcal{G}(\omega(G) + 1, \mathrm{MAIS}(G) + 1) \leq Q(\omega(G) + 1, \mathrm{MAIS}(G) + 1) \leq n,$$

which implies that G has a clique of size larger than $\omega(G)$ or an acyclic induced sub-digraph of size larger than $\mathrm{MAIS}(G)$, in contradiction. Now, by the definition of the function f_Q, it follows from (4) that $f_Q(n) \leq \omega(G) \cdot \mathrm{MAIS}(G)$, hence there exists a clique of G of size at least $\frac{f_Q(n)}{\mathrm{MAIS}(G)}$. This allows us to apply Lemma 2.1 and to obtain that for every digraph $G \in \mathcal{G}$ on n vertices, it holds that

$$\mathrm{CC}(G) \leq \left(\sum_{i=1}^{n} \frac{1}{f_Q(i)} \right) \cdot \mathrm{MAIS}(G). \tag{5}$$

For the algorithmic statement of the theorem, suppose that there exists a polynomial-time algorithm that given a digraph $G \in \mathcal{G}$ on n vertices finds a clique C of G and a set I for which $G[I]$ is acyclic, such that for all integers $s, t \geq 1$, if $n \geq Q(s,t)$ then either $|C| \geq s$ or $|I| \geq t$. Fix a digraph $G \in \mathcal{G}$ on n vertices. While running on G, the sets C and I returned by the given algorithm must

satisfy $Q(|C| + 1, |I| + 1) > n$, hence $f_Q(n) \leq |C| \cdot |I|$. It thus follows that the clique C returned by the algorithm satisfies $|C| \geq \frac{f_Q(n)}{|I|} \geq \frac{f_Q(n)}{\text{MAIS}(G)}$. Therefore, using the algorithmic statement of Lemma 2.1, it follows that there exists a polynomial-time algorithm that given a digraph $G \in \mathcal{G}$ on n vertices finds a clique cover of G whose size does not exceed the bound in (5). This completes the proof. ∎

2.1 General digraphs

Our first application of Theorem 1.3 concerns the family of all digraphs. Namely, we prove Theorem 1.4, which says that there exists a polynomial-time algorithm that given a digraph G on n vertices finds a clique cover of G of size $O(\frac{n}{\log n}) \cdot \text{MAIS}(G)$. To do so, we need the following lemma, whose proof relies on a directed variant of an argument from Ramsey theory due to Erdős and Szekeres [17].

Lemma 2.2. *Let* $Q : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ *be the function defined by* $Q(s,t) = \binom{s+t-2}{s-1} \cdot 2^{t-1}$. *There exists a polynomial-time algorithm that given a digraph G on n vertices finds a clique C of G and a set I for which $G[I]$ is acyclic, such that for all integers $s, t \geq 1$, if $n \geq Q(s,t)$ then either $|C| \geq s$ or $|I| \geq t$.*

Proof: We define a recursive algorithm, called DRamsey, that given a digraph $G = (V, E)$ returns a pair (C, I) of subsets of V. The algorithm is defined as follows.

1. If $V = \emptyset$, then return the pair $(\emptyset, \emptyset)$.

2. Choose an arbitrary vertex $u \in V$.

3. $(C_1, I_1) \leftarrow \text{DRamsey}(G[V_1])$, where $V_1 = \{v \in V \mid (u,v) \in E \text{ and } (v,u) \in E\}$.

4. $(C_2, I_2) \leftarrow \text{DRamsey}(G[V_2])$, where $V_2 = \{v \in V \mid (u,v) \notin E \text{ and } (v,u) \in E\}$.

5. $(C_3, I_3) \leftarrow \text{DRamsey}(G[V_3])$, where $V_3 = V \setminus (V_1 \cup V_2 \cup \{u\})$.

6. Let C be a largest set among $C_1 \cup \{u\}$, C_2, and C_3, let I be a largest set among I_1, $I_2 \cup \{u\}$, and $I_3 \cup \{u\}$, and return the pair (C, I).

Observe that for any choice of a vertex u in Step 2 of the DRamsey algorithm, the sets V_1, V_2, and V_3 defined in Steps 3, 4, and 5 do not include u and are pairwise disjoint. This implies that throughout the run of the algorithm on a digraph G, every recursive call chooses in Step 2 a distinct vertex of G, hence the total number of recursive calls does not exceed the number of vertices in G. Since the number of operations made by the algorithm for preparing the three inputs of the recursive calls and for combining the returned pairs to produce the output of the algorithm is polynomial, we conclude that the algorithm can be implemented in polynomial time.

We first prove that on every input digraph $G = (V, E)$, the DRamsey algorithm returns a pair (C, I) such that C is a clique of G and $G[I]$ is acyclic. This is shown by induction on the number of vertices of G. If G has no vertices, this trivially holds by Step 1 of the algorithm. Otherwise, the algorithm chooses in Step 2 an arbitrary vertex $u \in V$, and for each $i \in [3]$, it calls the DRamsey algorithm recursively on the sub-digraph $G[V_i]$ to obtain a pair (C_i, I_i), where V_1, V_2, V_3 are defined

9

in Steps 3, 4, and 5 of the algorithm. By the inductive assumption, for each $i \in [3]$, C_i is a clique of $G[V_i]$, and I_i induces an acyclic sub-digraph of $G[V_i]$.

By the definition of V_1, the vertex u is connected to all vertices of V_1 with edges in both directions, hence $C_1 \cup \{u\}$ is a clique of G. It thus follows that the set C, defined in Step 6 as a largest set among $C_1 \cup \{u\}$, C_2, and C_3, is a clique of G, as required. By the definition of V_2, the vertex u has no directed edges to the vertices of V_2. Since I_2 induces an acyclic sub-digraph of G, it follows that $I_2 \cup \{u\}$ induces such a sub-digraph as well. Similarly, by the definition of V_3, no vertex of V_3 has a directed edge to the vertex u. Since I_3 induces an acyclic sub-digraph of G, it follows that $I_3 \cup \{u\}$ induces such a sub-digraph as well. It thus follows that the set I, defined in Step 6 as a largest set among I_1, $I_2 \cup \{u\}$, and $I_3 \cup \{u\}$, induces an acyclic sub-digraph of G, as required.

We finally analyze the sizes of the sets returned by the DRamsey algorithm. To do so, for every pair of integers $s, t \geq 1$, let $\vec{P}(s, t)$ denote the smallest integer p such that on every digraph on at least p vertices, the DRamsey algorithm is guaranteed to return a clique of size at least s or a set of vertices that induces an acyclic sub-digraph of size at least t. We claim that for all $s, t \geq 2$, it holds that

$$\vec{P}(s, t) \leq \vec{P}(s-1, t) + 2 \cdot \vec{P}(s, t-1) - 2. \tag{6}$$

To see this, consider a digraph G on $p \geq \vec{P}(s-1, t) + 2 \cdot \vec{P}(s, t-1) - 2$ vertices. While running on G, the DRamsey algorithm calls itself recursively on three sub-digraphs of G, induced by three sets V_1, V_2, and V_3, which satisfy $|V_1| + |V_2| + |V_3| = p - 1$. Our assumption on p implies that either $|V_1| \geq \vec{P}(s-1, t)$, or $|V_2| \geq \vec{P}(s, t-1)$, or $|V_3| \geq \vec{P}(s, t-1)$. By the definition of $\vec{P}(s, t)$, the three pairs (C_1, I_1), (C_2, I_2), and (C_3, I_3) returned by the recursive calls satisfy $|C_1| \geq s-1$ or $|I_1| \geq t$ in the first case, $|C_2| \geq s$ or $|I_2| \geq t-1$ in the second case, and $|C_3| \geq s$ or $|I_3| \geq t-1$ in the third. By the definition of the algorithm, it follows that in each of these cases, the DRamsey algorithm returns on G a clique of size at least s or a set of vertices that induces an acyclic sub-digraph of size at least t, as required.

It remains to verify that $\vec{P}(s, t) \leq Q(s, t)$ for all $s, t \geq 1$. This can be checked by induction on $s + t$. First, if either $s = 1$ or $t = 1$, then $\vec{P}(s, t) = 1 \leq Q(s, t)$. Next, for the induction step, combine (6) with the inductive assumption on $s + t - 1$ to obtain that

$$
\begin{aligned}
\vec{P}(s, t) \quad &\leq \quad \vec{P}(s-1, t) + 2 \cdot \vec{P}(s, t-1) - 2 \\
&\leq \quad Q(s-1, t) + 2 \cdot Q(s, t-1) \\
&= \quad \binom{s+t-3}{s-2} \cdot 2^{t-1} + 2 \cdot \binom{s+t-3}{s-1} \cdot 2^{t-2} \\
&= \quad \left(\binom{s+t-3}{s-2} + \binom{s+t-3}{s-1} \right) \cdot 2^{t-1} \\
&= \quad \binom{s+t-2}{s-1} \cdot 2^{t-1} = Q(s, t),
\end{aligned}
$$

where the third equality holds by Pascal's identity. This completes the proof. ∎

Now, we combine Theorem 1.3 with Lemma 2.2 to prove Theorem 1.4.

Proof of Theorem 1.4: Let $Q : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ be the function given in Lemma 2.2, defined by $Q(s,t) = \binom{s+t-2}{s-1} \cdot 2^{t-1}$. Notice that for all integers $s, t \geq 1$, it holds that

$$Q(s+1, t+1) = \binom{s+t}{s} \cdot 2^t \leq 2^{s+t} \cdot 2^t = 2^{s+2t}.$$

It follows, for every integer n, that if $Q(s+1, t+1) > n$ then $2^{s+2t} > n$, which implies that either $s > \lfloor (\log_2 n)/3 \rfloor$ or $t > \lfloor (\log_2 n)/3 \rfloor$. This yields that the function f_Q associated with Q in Theorem 1.3 satisfies

$$f_Q(n) > \lfloor (\log_2 n)/3 \rfloor. \tag{7}$$

Equipped with the algorithm given by Lemma 2.2, we apply Theorem 1.3 to the family of all digraphs. It follows that there exists a polynomial-time algorithm that given a digraph G on n vertices finds a clique cover of G whose size does not exceed

$$\Big(\sum_{i=1}^{n} \frac{1}{f_Q(i)} \Big) \cdot \mathrm{MAIS}(G) \leq O\Big(\frac{n}{\log n} \Big) \cdot \mathrm{MAIS}(G),$$

where the above inequality can be verified using (7). This completes the proof. ∎

2.2 Digraph families with polynomial Ramsey numbers

Our next application of Theorem 1.3 concerns digraph families whose Ramsey numbers grow polynomially.

Theorem 2.3. *Let $\mathcal{G}$ be a hereditary family of digraphs such that for some constants c and $a \geq 1$, it holds that $\vec{R}_{\mathcal{G}}(s,t) \leq c \cdot (s \cdot t)^a$ for all integers $s, t \geq 1$. Then for every digraph $G \in \mathcal{G}$ on n vertices, it holds that $\mathrm{CC}(G) \leq h(n) \cdot \mathrm{MAIS}(G)$ for a function $h : \mathbb{N} \rightarrow \mathbb{N}$ satisfying $h(n) = \Theta(\log n)$ if $a = 1$, and $h(n) = \Theta(n^{1-1/a})$ otherwise.*

Proof: Let $\mathcal{G}$ be a hereditary family of digraphs as in the theorem. Let $Q : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ be the function defined by $Q(s,t) = \vec{R}_{\mathcal{G}}(s,t)$ for all s, t. Notice that for all integers $s, t \geq 1$, it holds that

$$Q(s+1, t+1) \leq c \cdot (s+1)^a \cdot (t+1)^a \leq c \cdot (2s)^a \cdot (2t)^a.$$

It therefore follows, for every integer n, that if $Q(s+1, t+1) > n$ then $s \cdot t > n^{1/a}/(4 \cdot c^{1/a})$. This yields that the function f_Q associated with Q in Theorem 1.3 satisfies $f_Q(n) \geq \Omega(n^{1/a})$. Applying Theorem 1.3 to the family $\mathcal{G}$, it follows that for every digraph $G \in \mathcal{G}$ on n vertices, it holds that $\mathrm{CC}(G) \leq h(n) \cdot \mathrm{MAIS}(G)$ for a function $h : \mathbb{N} \rightarrow \mathbb{N}$ satisfying $h(n) = \Theta(\sum_{i=1}^{n} \frac{1}{i^{1/a}})$. A standard integral calculation implies that for $a = 1$, it holds that $h(n) = \Theta(\log n)$, and for $a > 1$, it holds that $h(n) = \Theta(n^{1-1/a})$. This completes the proof. ∎

2.3 Graph families

Our method for finding clique covers of digraphs can also be used in the undirected setting. To see this, for any given hereditary family $\mathcal{G}$ of graphs, apply Theorem 1.3 to the family of digraphs

11

obtained from the graphs of $\mathcal{G}$ by replacing every edge by two oppositely directed edges. Observe that this operation preserves the cliques of each graph of $\mathcal{G}$ and, in addition, it transforms the notion of independent sets of the graph to the notion of acyclic induced sub-digraphs of the corresponding digraph. This allows us to derive the following undirected variant of Theorem 1.3.

Theorem 2.4. *Let $\mathcal{G}$ be a hereditary family of graphs, and let $Q : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be a function such that $R_{\mathcal{G}}(s,t) \leq Q(s,t)$ for all integers $s,t \geq 1$. Let $f_Q : \mathbb{N} \to \mathbb{N}$ denote the function defined by*

$$f_Q(n) = \min_{s,t \in \mathbb{N}} \{s \cdot t \mid Q(s+1,t+1) > n\}.$$

Then for every graph $G \in \mathcal{G}$ on n vertices, it holds that

$$\overline{\chi}(G) \leq \Big(\sum_{i=1}^{n} \frac{1}{f_Q(i)} \Big) \cdot \alpha(G). \tag{8}$$

Suppose further that there exists a polynomial-time algorithm that given a graph $G \in \mathcal{G}$ on n vertices finds a clique C of G and an independent set I of G, such that for all integers $s,t \geq 1$, if $n \geq Q(s,t)$ then either $|C| \geq s$ or $|I| \geq t$. Then there exists a polynomial-time algorithm that given a graph $G \in \mathcal{G}$ on n vertices finds a clique cover of G whose size does not exceed the bound in (8).

Theorem 2.4 can be used to derive the algorithmic result of Boppana and Halldórsson [10], stated earlier as Theorem 1.1. For completeness, we provide the details in Appendix A. We further state here the following corollary of Theorem 2.3 for the undirected setting.

Corollary 2.5. *Let $\mathcal{G}$ be a hereditary family of graphs such that for some constants c and $a \geq 1$, it holds that $R_{\mathcal{G}}(s,t) \leq c \cdot (s \cdot t)^a$ for all integers $s,t \geq 1$. Then for every graph $G \in \mathcal{G}$ on n vertices, it holds that $\overline{\chi}(G) \leq h(n) \cdot \alpha(G)$ for a function $h : \mathbb{N} \to \mathbb{N}$ satisfying $h(n) = \Theta(\log n)$ if $a = 1$, and $h(n) = \Theta(n^{1-1/a})$ otherwise.*

It was shown in [3, Theorem 1] that for graphs on n vertices taken from a hereditary family $\mathcal{G}$ whose Ramsey numbers satisfy $R_{\mathcal{G}}(s,t) \leq O(s^a \cdot t^b)$ for constants $a,b \geq 1$, the ratio between the clique cover and independence numbers is bounded by $O(n^{1-1/(a+b+1)})$. Our Corollary 2.5 improves this bound to $O(n^{1-1/\max(a,b)})$ when $\max(a,b) > 1$, and to $O(\log n)$ when $a = b = 1$.

3 Clique Covering of Quasi-Line Graphs

A graph G is called a quasi-line graph if the neighborhood of every vertex of G can be partitioned into two cliques. Note that all line graphs are quasi-line graphs, and that all quasi-line graphs are claw-free, however, the converse of each of these statements is false (see [13]). We present here the simple proof of Theorem 1.5, which asserts that there exists a polynomial-time algorithm that given a quasi-line graph G finds a clique cover of G of size at most $2 \cdot \alpha(G)$. As mentioned earlier, the theorem strengthens a result on line graphs given in [14, Proposition 5].

Proof of Theorem 1.5: Consider the algorithm that given a quasi-line graph G acts as follows. The algorithm maintains a collection $\mathcal{C}$ of cliques of G initiated as $\mathcal{C} = \emptyset$. As long as G has at least one vertex, the algorithm chooses an arbitrary vertex u in G, adds to $\mathcal{C}$ at most two cliques of G that

cover the closed neighborhood of u, and removes their vertices from G. Finally, when G has no vertices, the algorithm returns the obtained collection $\mathcal{C}$.

We first show that the algorithm is guaranteed to find a clique cover of the input graph in polynomial time. Let G be a quasi-line graph. By definition, the subgraph induced by the neighborhood of any vertex u can be partitioned into at most two cliques of G, which can be found in polynomial time (say, by 2-coloring the complement of the subgraph induced by the neighborhood of u). By adding u to one of those cliques, we obtain the cliques that cover the closed neighborhood of u, as needed in every iteration of the algorithm. Since the family of quasi-line graphs is hereditary, we are allowed to repeat this process after removing the covered vertices from G. Once no vertices remain in the graph, the collection $\mathcal{C}$ forms a clique cover of G. Since the number of vertices of G decreases in every iteration, the number of iterations does not exceed the number of vertices of the input graph, hence the total running time is polynomial.

We next analyze the size of the clique cover returned by the algorithm. For a quasi-line graph G, let r denote the number of iterations applied by the above algorithm while running on G, and let $u_1, \ldots, u_r$ denote the vertices chosen throughout these iterations. Since every iteration adds to the collection $\mathcal{C}$ at most two cliques, it follows that the returned $\mathcal{C}$ satisfies $|\mathcal{C}| \leq 2 \cdot r$. We further observe that the vertices $u_1, \ldots, u_r$ form an independent set in G. This indeed holds because whenever a vertex is chosen by the algorithm, its closed neighborhood is removed from the graph. Therefore, the clique cover $\mathcal{C}$ returned by the algorithm satisfies $|\mathcal{C}| \leq 2 \cdot r \leq 2 \cdot \alpha(G)$, as required. ∎

Remark 3.1. *The guarantee of the algorithm given by Theorem 1.5 is essentially optimal, even for line graphs. To see this, let G denote the line graph of the complete graph on n vertices for an integer $n \geq 4$. Consequently, the vertices of G are all the 2-subsets of $[n]$, where two such subsets are adjacent if they have a non-trivial intersection. It follows that G is the complement of the Kneser graph $K(n, 2)$, hence it holds that $\alpha(G) = \lfloor n/2 \rfloor$ and $\overline{\chi}(G) = n - 2$ (see [23]).*

4   Generalized Index Coding

In this section, we consider a generalized version of the index coding problem, studied in [9] and defined as follows. An instance H of the problem consists of integers m and n such that $m \geq n$, and for each $j \in [m]$, an element $r(j) \in [n]$ and a set $N(j) \subseteq [n] \setminus \{r(j)\}$. Here, a sender holds an n-symbol message $x \in \Sigma^n$ over some alphabet Σ, and m denotes the number of receivers. The jth receiver is interested in the symbol $x_{r(j)}$ of x and has the restriction $x_{N(j)}$ of x to the indices of $N(j)$ as side information. We naturally assume that for each $i \in [n]$ there exists some $j \in [m]$ with $r(j) = i$. An index code for H over Σ is an encoding function $E : \Sigma^n \to \Sigma_E$ for some alphabet Σ_E, such that for each $j \in [m]$, there exists a decoding function $g_j : \Sigma_E \times \Sigma^{|N(j)|} \to \Sigma$ satisfying $g_j(E(x), x_{N(j)}) = x_{r(j)}$ for all $x \in \Sigma^n$. The encoding length in bits of this index code is $\lceil \log_2 |\Sigma_E| \rceil$. For an integer $t \geq 1$, we let $\beta_t(H)$ denote the smallest possible encoding length in bits of an index code for H over an alphabet Σ of size $|\Sigma| = 2^t$. The index coding rate of H, denoted by $\beta(H)$, is defined by $\beta(H) = \lim_{t \to \infty} \frac{\beta_t(H)}{t}$. Note that the special case of the problem with $m = n$ coincides with the standard index coding problem on digraphs. Note further that $\beta(H) \leq \beta_1(H) \leq n$, where the second inequality follows by the trivial index code that transmits the entire message.

Let $\mathcal{C}$ be a collection of subsets of $[n]$, such that for each $j \in [m]$, there exists a set $C \in \mathcal{C}$ satisfying $r(j) \in C$ and $C \setminus \{r(j)\} \subseteq N(j)$. Consider the encoding function over $\Sigma = \{0,1\}$ that for every set $C \in \mathcal{C}$ includes the xor of the bits associated with C. This encoding function forms an index code for H, because the jth receiver is able to discover its required bit given its side information and the xor associated with a set $C \in \mathcal{C}$ which satisfies $r(j) \in C$ and $C \setminus \{r(j)\} \subseteq N(j)$. It thus follows that for such a collection $\mathcal{C}$, it holds that $\beta(H) \leq \beta_1(H) \leq |\mathcal{C}|$. We refer to such an index code for H, in this generalized setting, as a clique cover index code.

We next mention a lower bound from [9] on the index coding rate in the generalized setting. An expanding sequence of size k for H is a sequence $j_1, \ldots, j_k \in [m]$ such that for each $\ell \in [k]$, it holds that $r(j_\ell) \notin \cup_{t < \ell} N(j_t)$, that is, the symbol required by receiver j_ℓ is unknown to receivers $j_1, \ldots, j_{\ell-1}$. Let $\mathrm{MES}(H)$ denote the maximum size of an expanding sequence for H. It was proved in [9, Lemma III.1] that for every instance H of the generalized index coding problem, it holds that $\beta(H) \geq \mathrm{MES}(H)$.

We prove the following algorithmic result for the generalized index coding problem. Note that its guarantee is useful only for instances whose number of receivers m satisfies $m = o(n \cdot \log n)$.

Theorem 4.1. *There exists a polynomial-time algorithm that given an instance H of the generalized index coding problem with n symbols and m receivers ($m \geq n$) finds a clique cover index code for H of length* $O(\frac{m}{\log n}) \cdot \mathrm{MES}(H) \leq O(\frac{m}{\log n}) \cdot \beta(H)$.

Proof: Let H be an instance of the generalized index coding problem with n symbols and m receivers ($m \geq n$). As above, for each $j \in [m]$, let $r(j) \in [n]$ denote the index of the symbol required by the jth receiver, and let $N(j) \subseteq [n] \setminus \{r(j)\}$ denote the set of indices of the symbols that the jth receiver has as side information.

Consider the algorithm that given such an instance H maintains a set $A_i \subseteq [m]$ for each $i \in [n]$, initiated as the set of indices of the receivers that are interested in the ith symbol, namely, as $\{j \in [m] \mid r(j) = i\}$. The ith symbol is called active whenever $A_i \neq \emptyset$. The algorithm chooses for every active symbol i some receiver $j_i \in A_i$ and constructs an instance $G = (V, E)$ of the (standard) index coding problem as follows. The vertex set V of G consists of the indices of the currently active symbols of H, and each vertex i has a directed edge to the vertices of $N(j_i) \cap V$, i.e., the indices of the active symbols known to receiver j_i. Note that an index code for G satisfies all the receivers j_i with $i \in V$. Our algorithm calls the algorithm given by Theorem 1.7 on G to obtain a clique cover index code for G of length

$$O\left(\frac{|V|}{\log |V|}\right) \cdot \mathrm{MAIS}(G) \leq O\left(\frac{|V|}{\log |V|}\right) \cdot \mathrm{MES}(H), \tag{9}$$

where the inequality holds because the receivers associated with an acyclic induced sub-digraph of G, ordered appropriately, form an expanding sequence of H.

The algorithm proceeds by removing the receiver j_i from A_i for each $i \in V$ and repeating the process as long as there are at least, say, $\sqrt{n}$ active symbols. Once there are fewer than this number of active symbols, the algorithm constructs the trivial index code that transmits all the active symbols, satisfying all the remaining receivers of $\cup_{i \in [n]} A_i$. Note that the length of the latter is smaller than $\sqrt{n}$. Observe that the index code defined as the concatenation of all the index codes constructed by our algorithm is a valid clique cover index code for H. Observe further that the

14

algorithm runs in polynomial time, because in each iteration the number of receivers decreases, and because the algorithm from Theorem 1.7 runs in polynomial time.

It remains to analyze the length of the index code produced by the algorithm. For an instance H with n symbols and m receivers, let $G_1 = (V_1, E_1), \ldots, G_t = (V_t, E_t)$ denote the instances of the (standard) index coding problem constructed throughout the run of the algorithm. By the definition of the algorithm we have $|V_i| \geq \sqrt{n}$ for all $i \in [t]$. Using (9), this implies that for each $i \in [t]$, the length of the index code produced for G_i is bounded by

$$O\Big(\frac{|V_i|}{\log |V_i|}\Big) \cdot \mathrm{MES}(H) \leq O\Big(\frac{|V_i|}{\log n}\Big) \cdot \mathrm{MES}(H).$$

Since the vertices of the digraphs $G_1, \ldots, G_t$ represent distinct receivers of H, their vertex sets satisfy $\sum_{i=1}^{t} |V_i| \leq m$. Recalling that the last component of the constructed index code has length smaller than $\sqrt{n}$, we obtain that the total length of the index code produced by the algorithm does not exceed

$$O\Big(\sum_{i=1}^{t} \frac{|V_i|}{\log n}\Big) \cdot \mathrm{MES}(H) + \sqrt{n} \leq O\Big(\frac{m}{\log n}\Big) \cdot \mathrm{MES}(H),$$

where the inequality holds using $m \geq n$. This completes the proof. ∎

Acknowledgments

References

[1] N. Alon, E. Lubetzky, U. Stav, A. Weinstein, and A. Hassidim. Broadcasting with side information. In *Proc. of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS'08)*, pages 823–832, 2008.

[2] N. Alon and J. H. Spencer. *The Probabilistic Method*. Wiley Publishing, 4th edition, 2016.

[3] F. Arbabjolfaei and Y. Kim. Approximate capacity of index coding for some classes of graphs. In *Proc. of the IEEE International Symposium on Information Theory (ISIT'16)*, pages 2154–2158, 2016.

[4] F. Arbabjolfaei and Y. Kim. Fundamentals of index coding. *Found. Trends Commun. Inf. Theory*, 14(3–4):163–346, 2018.

[5] P. Awasthi, M. Charikar, R. Krishnaswamy, and A. K. Sinop. The hardness of approximation of Euclidean k-means. In *Proc. of the 31st International Symposium on Computational Geometry (SoCG'15)*, pages 754–767, 2015.

[6] Z. Bar-Yossef, Y. Birk, T. S. Jayram, and T. Kol. Index coding with side information. *IEEE Trans. Inform. Theory*, 57(3):1479–1494, 2011. Preliminary version in FOCS'06.

[7] R. Belmonte, P. Heggernes, P. van 't Hof, A. Rafiey, and R. Saei. Graph classes and Ramsey numbers. *Discret. Appl. Math.*, 173:16–27, 2014.

[8] Y. Birk and T. Kol. Coding on demand by an informed source (ISCOD) for efficient broadcast of different supplemental data to caching clients. *IEEE Trans. Inform. Theory*, 52(6):2825–2830, 2006. Preliminary version in INFOCOM'98.

[9] A. Blasiak, R. Kleinberg, and E. Lubetzky. Broadcasting with side information: Bounding and approximating the broadcast rate. *IEEE Trans. Inform. Theory*, 59(9):5811–5823, 2013.

[10] R. B. Boppana and M. M. Halldórsson. Approximating maximum independent sets by excluding subgraphs. *BIT Numer. Math.*, 32(2):180–196, 1992. Preliminary version in SWAT'90.

[11] D. Chawin and I. Haviv. Improved NP-hardness of approximation for orthogonality dimension and minrank. *SIAM J. Discret. Math.*, 37(4):2670–2688, 2023. Preliminary version in STACS'23.

[12] D. Chawin and I. Haviv. Hardness of linear index coding on perturbed instances. *IEEE Trans. Inform. Theory*, 70(2):1388–1396, 2024. Preliminary version in Allerton'22.

[13] M. Chudnovsky and P. D. Seymour. The structure of claw-free graphs. In B. S. Webb, editor, *Surveys in Combinatorics 2005*, volume 327 of *London Mathematical Society Lecture Note Series*, pages 153–172. Cambridge University Press, 2005.

[14] H. R. Daneshpajouh, F. Meunier, and G. Mizrahi. Colorings of complements of line graphs. *J. Graph Theory*, 98(2):216–233, 2021.

[15] P. Erdős. Some remarks on chromatic graphs. *Colloq. Math.*, 16:253–256, 1967.

[16] P. Erdős and L. Moser. On the representation of directed graphs as unions of orderings. *Magyar Tud. Akad. Mat. Kutató Int. Közl.*, 9:125–132, 1964.

[17] P. Erdős and G. Szekeres. A combinatorial problem in geometry. *Compositio Mathematica*, 2:463–470, 1935.

[18] U. Feige. Approximating maximum clique by removing subgraphs. *SIAM J. Discret. Math.*, 18(2):219–225, 2004.

[19] W. H. Haemers. An upper bound for the Shannon capacity of a graph. In L. Lovász and V. T. Sós, editors, *Algebraic Methods in Graph Theory*, volume 25/I of *Colloquia Mathematica Societatis János Bolyai*, pages 267–272. Bolyai Society and North-Holland, 1978.

[20] M. M. Halldórsson. A still better performance guarantee for approximate graph coloring. *Inf. Process. Lett.*, 45(1):19–23, 1993.

[21] J. Körner and G. Simonyi. A Sperner-type theorem and qualitative independence. *J. Comb. Theory, Ser. A*, 59(1):90–103, 1992.

[22] M. Langberg and A. Sprintson. On the hardness of approximating the network coding capacity. *IEEE Trans. Inform. Theory*, 57(2):1008–1014, 2011. Preliminary version in ISIT'08.

[23] L. Lovász. Kneser's conjecture, chromatic number, and homotopy. *J. Comb. Theory, Ser. A*, 25(3):319–324, 1978.

[24] C. E. Shannon. The zero error capacity of a noisy channel. *Institute of Radio Engineers, Trans. Inform. Theory*, IT-2:8–19, 1956.

[25] J. H. van Lint and R. M. Wilson. *A course in combinatorics*. Cambridge University Press, second edition, 2001.

[26] D. Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory of Computing*, 3(6):103–128, 2007. Preliminary version in STOC'06.

A Proof of Theorem 1.1

We show here that the algorithmic result of Boppana and Halldórsson [10], stated as Theorem 1.1, can be derived from our Theorem 2.4. The proof requires the following lemma, which is given implicitly in [10] and relies on an argument of [17].

Lemma A.1. *Let $Q : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be the function defined by $Q(s,t) = \binom{s+t-2}{s-1}$. There exists a polynomial-time algorithm that given a graph G on n vertices finds a clique C of G and an independent set I of G, such that for all integers $s, t \geq 1$, if $n \geq Q(s,t)$ then either $|C| \geq s$ or $|I| \geq t$.*

Proof: We define a recursive algorithm, called Ramsey, that given a graph $G = (V, E)$ returns a pair (C, I) of subsets of V. The algorithm is defined as follows.

1. If $V = \emptyset$, then return the pair $(\emptyset, \emptyset)$.

2. Choose an arbitrary vertex $u \in V$.

3. $(C_1, I_1) \leftarrow \mathsf{Ramsey}(G[V_1])$, where $V_1 = \{v \in V \mid \{u, v\} \in E\}$.

4. $(C_2, I_2) \leftarrow \mathsf{Ramsey}(G[V_2])$, where $V_2 = V \setminus (V_1 \cup \{u\})$.

5. Let C be a largest set among $C_1 \cup \{u\}$ and C_2, let I be a largest set among I_1 and $I_2 \cup \{u\}$, and return the pair (C, I).

Observe that for any choice of a vertex u in Step 2 of the Ramsey algorithm, the sets V_1 and V_2 defined in Steps 3 and 4 do not include u and are disjoint. This implies that throughout the run of the algorithm on a graph G, every recursive call chooses in Step 2 a distinct vertex of G. It follows that the algorithm can be implemented in polynomial time.

We first prove that on every input graph $G = (V, E)$, the Ramsey algorithm returns a pair (C, I) such that C is a clique of G and I is an independent set of G. This is shown by induction on the number of vertices of G. If G has no vertices, this trivially holds by Step 1 of the algorithm. Otherwise, the algorithm chooses in Step 2 an arbitrary vertex $u \in V$, and for each $i \in [2]$, it calls the Ramsey algorithm recursively on the subgraph $G[V_i]$ to obtain a pair (C_i, I_i), where V_1, V_2 are defined in Steps 3 and 4 of the algorithm. By the inductive assumption, for each $i \in [2]$, C_i is a clique of $G[V_i]$, and I_i is an independent set of $G[V_i]$.

By the definition of V_1, the vertex u is adjacent to all vertices of V_1, hence $C_1 \cup \{u\}$ is a clique of G. It thus follows that the set C, defined in Step 5 as a largest set among $C_1 \cup \{u\}$ and C_2, is a clique of G, as required. By the definition of V_2, the vertex u has no edges to the vertices of V_2,

hence $I_2 \cup \{u\}$ is an independent set of G. It thus follows that the set I, defined in Step 5 as a largest set among I_1 and $I_2 \cup \{u\}$, is an independent set of G, as required.

We finally analyze the sizes of the sets returned by the Ramsey algorithm. To do so, for every pair of integers $s, t \geq 1$, let $P(s,t)$ denote the smallest integer p such that on every graph on at least p vertices, the Ramsey algorithm is guaranteed to return a clique of size at least s or an independent set of size at least t. We claim that for all $s, t \geq 2$, it holds that

$$P(s,t) \leq P(s-1,t) + P(s,t-1) - 1. \tag{10}$$

To see this, consider a graph G on $p \geq P(s-1,t) + P(s,t-1) - 1$ vertices. While running on G, the Ramsey algorithm calls itself recursively on two subgraphs of G, induced by sets V_1 and V_2, which satisfy $|V_1| + |V_2| = p - 1$. Our assumption on p implies that either $|V_1| \geq P(s-1,t)$ or $|V_2| \geq P(s,t-1)$. By the definition of $P(s,t)$, the two pairs (C_1, I_1) and (C_2, I_2) returned by the recursive calls satisfy $|C_1| \geq s-1$ or $|I_1| \geq t$ in the first case, and $|C_2| \geq s$ or $|I_2| \geq t-1$ in the second case. By the definition of the algorithm, it follows that in each of these cases, the Ramsey algorithm returns on G a clique of size at least s or an independent set of size at least t, as required.

It remains to verify that $P(s,t) \leq Q(s,t)$ for all $s, t \geq 1$. This can be checked by induction on $s+t$. First, if either $s = 1$ or $t = 1$, then $P(s,t) = Q(s,t) = 1$. Next, for the induction step, combine (10) with the inductive assumption on $s+t-1$ to obtain that

$$
\begin{aligned}
P(s,t) \;&\leq\; P(s-1,t) + P(s,t-1) - 1 \\
&\leq\; Q(s-1,t) + Q(s,t-1) \\
&=\; \binom{s+t-3}{s-2} + \binom{s+t-3}{s-1} \\
&=\; \binom{s+t-2}{s-1} = Q(s,t),
\end{aligned}
$$

where the second equality holds by Pascal's identity. This completes the proof. ∎

Now, we combine Theorem 2.4 with Lemma A.1 to prove Theorem 1.1.

Proof of Theorem 1.1: Let $Q : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be the function given in Lemma A.1, defined by $Q(s,t) = \binom{s+t-2}{s-1}$. It can be shown (see, e.g., [15, Lemma 1]) that the function f_Q associated with Q in Theorem 2.4 satisfies

$$f_Q(n) \geq \Omega(\log^2 n). \tag{11}$$

Equipped with the algorithm given by Lemma A.1, we apply Theorem 2.4 to the family of all graphs. It follows that there exists a polynomial-time algorithm that given a graph G on n vertices finds a clique cover of G whose size does not exceed

$$\Big(\sum_{i=1}^{n} \frac{1}{f_Q(i)} \Big) \cdot \alpha(G) \leq O\Big(\frac{n}{\log^2 n} \Big) \cdot \alpha(G),$$

where the above inequality can be verified using (11). This completes the proof. ∎