Identifying Codes and Covering Problems

Moshe Laifenfeld and Ari Trachtenberg

Abstract

The *identifying code problem* for a given graph involves finding a minimum set of vertices whose neighborhoods uniquely overlap at any given graph vertex. Initially introduced in 1998, this problem has demonstrated its fundamental nature through a wide variety of applications, such as fault diagnosis, location detection, and environmental monitoring, in addition to deep connections to information theory, superimposed and covering codes, and tilings. This work establishes efficient reductions between the identifying code problem and the well-known set-covering problem, resulting in a tight hardness of approximation result and novel, provably tight polynomial-time approximations. The main results are also extended to r*-robust* identifying codes and analogous *set* $(2r+1)$-*multicover* problems. Finally, empirical support is provided for the effectiveness of the proposed approximations, including good constructions for well-known topologies such as infinite two-dimensional grids.

Index Terms

Identifying codes, robust identifying codes, hardness of approximation, set cover, test cover, distributed algorithms.

A version of this paper is scheduled to appear as:

- M. Laifenfeld and A. Trachtenberg, “Identifying Codes and Covering Problems”, IEEE Transactions on Information Theory, September 2008.

I. Introduction

An identifying code is a subset of vertices in a graph with the property that the (incoming) neighborhood of any vertex has a unique intersection with the code. For example, a three dimensional cube (as depicted in Figure 1) has a three-vertex identifying code (labeled $\{1,2,3\}$ in the figure). The neighborhood of each vertex in the graph intersects uniquely with this code, and such an intersection is called an *identifying set*[1]; given an identifying set, one can thus uniquely identify the vertex in the graph that produced it. In this case, the code provided is also optimal, because one needs at least $\lg 8 = 3$ code vertices to produce 8 distinct identifying sets (corresponding to the 8 vertices of the cube).[2] The goal of the *identifying code problem* is to find an identifying code of minimum cardinality for any given graph.

Identifying codes have been studied extensively since their introduction in 1998 [1], and they have formed a fundamental basis for a wide variety of theoretical work and practical applications.

This work was supported in part by the National Science Foundation under grants CCR-0133521, CNS-0435312 and CCF-0729158, and in part by DTRA grant HDTRA1-07-1-0004. The material was presented in part at the 44th Allerton Conference on Communication, Control, and Computing, September 2006.

The authors (moshel@bu.edu and trachten@bu.edu, respectively) are with the department of Electrical and Computer Engineering, Boston University, Boston MA 02215. The first author is also with the department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge MA 02139.

[1]Unlike traditional identifying codes, the empty set is considered a valid identifying set here.

[2]As is common, we use the notation $\lg(x)$ to denote $\log_2(x)$.

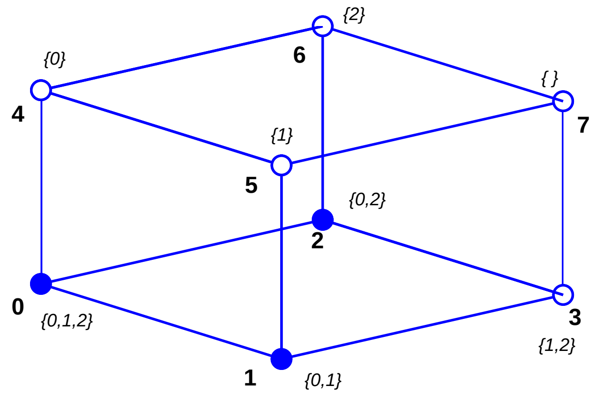


Fig. 1. Identifying code example. Solid vertices represent the code $\{1, 2, 3\}$. The unique intersections of the vertices neighborhoods with the code (identifying sets) appear in braces.

1) Applications: The initial application for identifying codes was to fault diagnosis in multiprocessor systems [1]. In this application, testers are positioned in the system according to an identifying code so that faults can be localized to a unique processor by considering only which testers detect faults within their neighborhood [1].

Identifying codes have since been extended and applied to location detection in hostile environments [2–4], to energy balancing of such systems [5], and to dynamic location detection agents [6]. In the first example, a coverage area is quantized into a finite number of clusters. In the mathematical model, the set of clusters corresponds to vertices in a graph, and edges represent averaged radio connectivity between clusters. Beacons are then placed in the clusters according to an identifying code, allowing a user traversing the coverage area to detect her location (cluster) according to the set of beacons she receives.

More recently, these codes were extended to applications for environmental monitoring [7], and joint monitoring and routing in wireless sensor networks [8]. In the former, sensors are placed in a subset of junctions of a utility network such as an air ventilation system, a water supply network, or a river bed, etc, in order to detect pollutants. Based on the timing and nature of the sensors' reports, a centralized processor is able to detect and determine the pollution source. The problem of placing the smallest number of sensors was found to be a variant of the identifying code problem [9]. For wireless networks, the identification property of identifying codes was used to uniquely label sensors in a network, providing natural means of routing on top of the traditional monitoring functionality [8].

2) Theoretical connections: From a theoretical perspective, identifying codes are closely linked to error-correcting codes, specifically, super-imposed codes [1, 10], covering codes [1, 6], and locating-dominating sets [11]. The intimate relationship between identifying codes and super-imposed codes was initially pointed out in [1] and further developed in [10, 12–14], providing optimal constructions and tight bounds on code size. Locating-dominating sets are very similar to identifying codes with the subtle difference that only the vertices not in the locating-dominating set are required to have unique identifying sets [15]. Unlike identifying codes, every graph admits a trivial locating-dominating set - the entire set of vertices. Links to other well-studied problems can be found in the literature: to the 3-satisfiability problem in [16], to the set cover problem in [9, 17], to the dominating-set and alarm placement problems in [18, 19], and to the set multicover and test cover problems in [9]. The test cover problem is, in fact, a generalization of the identifying code problem, and some of its results apply throughout.

Many variants of identifying codes have emerged since they were first introduced. In radius $(\rho > 1)$ identifying codes [1], the neighborhood of v is redefined to include all vertices at distance $\leq \rho$ from v. The $(1, \leq l)$-identifying codes [10, 13] can simultaneously identify any subset of at most l vertices. Dynamic identifying codes are identifying codes whose codewords form a walk through the graph. This variant was studied in [6] with applications to dynamic agents for fault detection in multiprocessor arrays. *Robust* identifying codes were suggested in [4] for applications of location detection in harsh environments, where vertices and connecting edges are likely to fail. Intuitively, an r-robust identifying code is a code that maintains its identification property in the event of a removal or insertion of up to r different vertices from all identifying sets. In the example of Figure 1, the set of all vertices forms a 1-robust identifying code for the cube. The observation that r-robust identifying codes are error correcting codes of minimum Hamming distance of $2r + 1$ was made in [4]. Theoretical bounds closely related to covering codes, and some efficient constructions for periodic geometries were further developed in [6]. Finally, the source identification problem, a variant through which the source of pollutant (traveling according to a given graph) is to be identified, has been shown to be NP-complete for both the general version [9] and a time-constrained version [7].

3) Approximating the optimal identifying code: In the most general situation, finding a minimum size identifying code for arbitrary undirected and directed graphs was proven to be NP-complete in [16, 20], based on a reduction from the 3-satisfiability problem [21]. An exception to this result is the specific case of directed [22] and undirected trees, for which there exists a polynomial-time algorithm for finding a minimum radius 1 identifying code.

Significant efforts in the research of identifying codes and their variants have focused on finding efficient constructions in two dimensional lattices, grids and Hamming spaces (see [12, 23–26], and [6] for a summary of recent results). Until recently, little has been published towards a polynomial time approximation algorithm for arbitrary graphs. In [2, 4] a polynomial-time greedy heuristic and its distributed variant were suggested for obtaining an identifying code in an arbitrary graph, and simulations showed it to work well over random graphs. Unfortunately, no guarantees for the quality of the obtained solution were presented, and Moncel later proved in [27] that no such guarantees exist.

Independently and in parallel, several groups have been looking into the question of approximability of identifying codes [9, 17, 18], obtaining polynomial-time approximations within an $O(\log |V|)$ factor of the optimal solution. In [18] the authors tied identifying codes to the dominating set problem, thereby showing that, under common complexity assumptions, approximating identifying codes within a sub-logarithmic factor is intractable. More precisely, it has been shown that identifying codes can be approximated within $O(\log |V|)$ factor, but they can not be approximated in polynomial time within $1 + \alpha \log |V|$ factor for some $\alpha > 0$. In our initial work [9], we have provided an explicit value for α by demonstrating that identifying codes are not approximable within a $\ln |V|$ factor unless NP $\subset$ DTIME$^{\log \log |V|}$; our result is based on a reduction from the set cover problem, and we use it to carry over the hardness result of Feige [28]. In this paper we further show that this bound is tight by adapting an algorithm developed by Berman et al. in [29] that attains this bound within a small additive constant. Using our reduction and with some additional work, other set cover hardness results (*e.g.,* [30]) may also be applied, obtaining related, but distinctly different, results. We also address the approximability of robust identifying codes by establishing a link to the set multi-cover problem.

A. Contributions

The main contribution of this work is to provide good polynomial time approximations to the identifying code problem, and to address the fundamental theoretical limits of such approximations. Specifically, we show that no polynomial-time algorithm can approximate identifying codes on *arbitrary graphs* within a $\ln |V|$ factor under commonly used complexity assumptions. Moreover, we show that a known test covering approximation [29] can be adapted to find identifying codes whose size is within a $1 + \ln |V|$ factor of optimal. The same fundamental questions were researched by others in parallel and independently [17, 18] providing similar, but weaker, results.

Our second contribution in this work is to provide good approximations to the robust identifying codes problem by tying it to the set multicover problem. Our approximation is guaranteed to produce robust identifying codes that are within a factor of 2 of the theoretical limit. We also develop two flavors of distributed algorithms that may be used for practical implementations in network applications.

B. Organization

The rest of the paper is organized as follows: We give formal definitions of the identifying code and the set cover problems in Section II. In Section III, we show a reduction from the set cover problem. In Section IV, we provide an $O(\log n)$-approximation algorithm for the identifying code problem, based on our reduction, and we show that this approximation ratio is tight. We then generalize this result in Section V, providing a hardness of approximation result for the identifying code problem, together with an approximation (based on [29]), which attains this bound up to a small additive constant. In Section VI, we discuss robust identifying codes and provide an approximation based on their relation to the set multi-cover problem. Finally in Section VI, we provide distributed implementations of our approximation algorithm, in addition to simulations results on random graphs and grids.

II. Formal definitions and related work

A. Identifying codes

Given a directed graph $G = (V, E)$, the *incoming ball* $B^{+}(v)$ consists of vertices that have an edge directed towards $v \in V$, together with v; likewise, the *outgoing ball* $B^{-}(v)$ consists of vertices that have an edge directed away from v, together with v. For undirected graphs, we shall simply use the notation $B(v) = B^{+}(v) = B^{-}(v)$.

As such, an identifying code is a set of vertices in a graph G with the property that any incoming ball in G has a unique intersection with the identifying code. More precisely, a non-empty subset $\mathbb{C} \subseteq V$ is called a *code* and its elements are *codewords*. For a given code $\mathbb{C}$, the *identifying set* $I_{\mathbb{C}}(v)$ of a vertex v is defined to be the codewords directed towards v, *i.e.*, $I_{\mathbb{C}}(v) = B^{+}(v) \cap \mathbb{C}$ (if $\mathbb{C}$ is not specified, it is assumed to be the set of all vertices V). A code $\mathbb{C}$ is thus an *identifying code* if each identifying set of the code is unique, or in other words $\forall u, v \in V \quad u = v \longleftrightarrow I_{\mathbb{C}}(u) = I_{\mathbb{C}}(v)$. Note that this definition does not include the standard assumption (which we will make in Section VI) that all identifying sets are non-empty.

1) Random graphs: Recently, random graphs and random geometric graphs were studied in the context of identifying codes [14, 31]. In [14] it was shown that for asymptotically large random graphs, any subset of a certain threshold size (logarithmic in the size of the graph) is almost surely an identifying code. It was also shown that the threshold is asymptotically sharp, *i.e.*, the probability of finding an identifying code of slightly smaller size asymptotically approaches zero. Unit disk geometric random graphs, in which vertices are placed on a two-dimensional plane and connected if their distance is less than some unit, were studied in [31]. There it was shown that, unlike large random graphs, most of the large unit-disk geometric random graphs do not possess identifying codes.

In contrast to very large random graphs, finding a minimum size identifying code for arbitrary undirected and directed graphs was proven to be NP-complete in [16, 20], based on a reduction from the 3-satisfiability problem.

2) Approximations: An initial attempt to develop a polynomial-time approximation was made in [2, 4]. Although the approximation worked well over random graphs it was later proven in [27] to have no general guarantees for the quality of the obtained solution. More recently, several groups have been independently looking into the question of approximability of identifying codes and dominating - locating sets [9, 17, 18], providing hardness of approximation results and polynomial time algorithms that approximate the optimal identifying code within a $O(\log |V|)$ factor.

B. Covering problems

1) Set cover: Let $\mathbf{U}$ be a base set of m elements and let $\mathcal{S}$ be a family of subsets[3] of $\mathbf{U}$. A *cover* $C \subseteq \mathcal{S}$ is a family of subsets whose union is $\mathbf{U}$. The *set cover problem* asks to find a cover C of smallest cardinality. The set cover problem is one of the oldest and most studied NP-hard problems [21]. It admits

[3]The term "family of subsets" is used to refer to a set of subsets

the following greedy approximation: at each step, and until exhaustion, choose the heretofore unselected set in $\mathcal{S}$ that covers the largest number of uncovered elements in the base set.

The performance ratio of the greedy set cover algorithm has also been well-studied. The classic results of Lovasz and Johnson [32,33] showed that $s_{\text{greedy}}/s_{\text{min}} = \Theta(\ln m)$, where s_{min} and s_{greedy} are the minimum and the greedy covers, and m is the size of the base set. Later Slavik [34] sharpened this ratio further, reaching a difference of less than 1.1 between the lower and upper bounds on the performance ratio. Recent studies on the hardness of approximation of the set cover problem can be found in [28,30]. Raz and Safra [30] showed that the set cover problem is NP-hard and that it can not be approximated by a polynomial algorithm within a $O(\log m)$ factor from an optimal solution unless P=NP. A tighter result was obtained by Feige [28] who showed that for any $\epsilon > 0$, no polynomial-time algorithm can approximate the minimum set cover within $(1-\epsilon)\ln m$ factor unless NP has deterministic algorithms operating in slightly super-polynomial time, *i.e.*, NP $\subset TIME\left[m^{O(\log\log m)}\right]$, suggesting that the greedy approach is one of the best polynomial approximations to the problem.

2) Multicover: The *minimum set k-multicover* problem is a natural generalization of the minimum set cover problem, in which one is given a pair $(\mathbf{U}, \mathcal{S})$ and seeks the smallest subset of $\mathcal{S}$ that covers every element in $\mathbf{U}$ at least k times (we defer more formal definitions to section VI). Often this problem is addressed as a special case of a more general family of integer optimization problems - the *covering integer problem* [35,36].

The set multicover problem admits a similar greedy heuristic to the set cover problem: in each iteration select the set which covers the maximum number of non k-multicovered elements. It is well known [36] that the performance guarantee of this heuristic is upper bounded by $1 + \log\alpha$, where α is the largest set's size.

3) Test cover: Another closely related problem is the *test cover problem*. This problem asks to find the smallest set $\mathbf{T}$ of given tests $T_i \subset \mathbf{U}$ such that any pair $x, y \in \mathbf{U}$ is differentiated by at least one test $T_i \in \mathbf{T}$ (*i.e.*, $|\{x,y\} \cap T_i| = 1$). The test covering problem appears naturally in identification problems, with roots in an agricultural study more than 20 years ago, regaining interest recently due to applications in bioinformatics [37,38].

Garey and Johnson [39] showed the test cover problem to be NP-hard and later Moret and Shapiro [40] suggested greedy approximations based on a reduction to the set cover problem. More recent work [37,38] studied different branch-and-bound approximations and established a hardness of approximation by extending the reduction in [40], and using a result of Feige [28]. Berman et al. [29] also suggested a novel greedy approximation and showed its performance ratio to be within a small constant from the hardness result of [38].

The test cover is clearly a general case of the identifying code problem, with tests corresponding to outgoing balls, and as such many of its results can be applied directly *e.g.,* [29]. Other results, such as the hardness of approximation, require some work due to the dependencies imposed by graph geography on nearby identifying sets. As such, the approach we use in Section IV bears some clear similarities to that of [37].

III. Identifying codes and the set cover problem

In this section, we establish a reduction between the identifying codes and the set cover problems. This reduction will serve as a basis for our subsequent approximation algorithms.

Formally, we connect the following problems:

a) SET-COVER:

`INSTANCE:`	Set S of subsets of a base set U.
`SOLUTION:`	A set $S' \subseteq S$ such that $\cup_{s \in S'} s = U$.
`MEASURE:`	The size of the cover: $|S'|$.

b) ID-CODE:

`INSTANCE:`	Graph $G = (V, E)$.
`SOLUTION:`	A set $C \subseteq V$ that is an identifying code of G.
`MEASURE:`	The size of the identifying code: $\lvert C\rvert$.

A. ID-CODE $\leq_P$ SET-COVER

We first show a reduction from the minimum identifying code problem to the set cover problem. We state the main theorem first and then we provide several definitions and lemmas that are used in its proof.

Theorem 1 *Given a graph G of n vertices, finding an identifying code requires no more computations than a set cover solution over a base set of $\frac{n(n-1)}{2}$ elements together with $O(n^3)$ operations (scalar multiplications, additions, or comparisons) on length n binary vectors.*

Intuitively, the reduction to a $(\mathbf{U}, \mathcal{S})$ set cover problem is established by setting $\mathbf{U}$ to contain all pairs of distinct vertices and $\mathcal{S} = \{\mathbf{S}_{v\in V}\}$ to be the set of $\mathbf{S}_v$ subsets that contain all pairs such that v is in the incoming ball of exactly one of them. We start with some notation and formal definitions to bootstrap the reduction.

Definition 1 *The* difference set $D(u,v)$ *is defined to be the symmetric difference between the incoming balls of vertices $u, v \in V$:*

$$\begin{aligned} D(u,v) &\doteq B^+(u) \oplus B^+(v) \\ &= \left[B^+(u) - B^+(v)\right] \cup \left[B^+(v) - B^+(u)\right], \end{aligned}$$

where subtraction denotes set difference. We shall also denote by $D_{\mathbb{C}}(u,v)$ the intersection of the code $\mathbb{C}$ with $D(u,v)$, namely $D_{\mathbb{C}}(u,v) = D(u,v) \cap \mathbb{C}$.

It is easy to see that $D_{\mathbb{C}}(u,v)$ is the symmetric difference between the identifying sets of vertices $u, v \in V$, namely

$$D_{\mathbb{C}}(u,v) \quad = \quad I_{\mathbb{C}}(u) \oplus I_{\mathbb{C}}(v).$$

We shall also use U to denote the set of all pairs of distinct vertices, i.e., $U = \{(u,z) | u \neq z \in V\}$.

Finally, the distinguishing set *of a vertex $v \in V$ is the set of vertex pairs (u,z) for which v is a member of their difference set:*

$$\delta_v = \{(u,z) \in U \,|\, v \in D(u,z)\}.$$

Note that the distinguishing set is independent of the code $\mathbb{C}$.

The following Lemma follows trivially from the definition of an identifying code.

Lemma 1 *A code $\mathbb{C}$ is an identifying code iff $\emptyset \notin \{D_{\mathbb{C}}(u,z) | (u,z) \in U\}$.*

Alternatively, we can define an identifying code in terms of distinguishing sets.

Lemma 2 $\mathbb{C}$ *is an identifying code iff the family of the distinguishing sets of its vertices covers $U = \{(u,z) \in V^2 \,|\, u \neq z\}$.*

Proof: From Lemma 1 for $\mathbb{C}$ to be an identifying code all difference sets should have at least one member. From the definition of distinguishing sets it then follows that for any $(u,v) \in U$ there exists some $c \in \mathbb{C}$ such that $(u,v) \in \delta_c$. Hence $\bigcup_{c\in\mathbb{C}} \delta_c = U$. The other direction follows similarly. ∎

Proof of Theorem 1: Consider the following construction of an identifying code.

ID$(G) \rightarrow \mathbb{C}$.

1) Compute the identifying sets $\{I(u) \,|\, u \in V\}$.

2) Compute the distinguishing sets $\Delta = \{\delta_u \,|\, u \in V\}$.
3) Compute $\mathbf{C} \leftarrow \mathrm{Minimum-Set-Cover}(U, \Delta)$.
4) Output $\mathbb{C} \leftarrow \{u \in V \,|\, \delta_u \in \mathbf{C}\}$, *i.e.*, vertices corresponding to distinguishing sets in the minimum cover.

The resulting code, $\mathbb{C}$, is guaranteed by Lemma 2 to be an identifying code, and the optimality of the set cover in step 3 guarantees that no smaller identifying code can be found. To complete the proof, we observe that computing the identifying sets $I(u)$ naively requires $\Theta(n^2)$ additions of binary vectors, and computing Δ requires n operations for each of the $\frac{n(n-1)}{2}$ elements in $|U|$. ∎

As a simple example of the construction behind Theorem 1, consider the graph in Figure 1. The identifying sets and distinguishing sets of the vertices are:

$v \in V$	$I(v)$	δ_v
0	{0,1,2,4}	{(0,3),(0,5),(0,6),(0,7),(1,3),(1,5),...}
1	{0,1,3,5}	{(0,2),(0,4),(0,6),(0,7),(1,2),(1,4),...}
2	{0,2,3,6}	{(0,1),(0,4),(0,6),(0,7),(1,0),(1,2),...}
3	{1,2,3,7}	
4	{0,4,5,6}	⋮
5	{1,4,5,7}	
6	{2,4,6,7}	
7	{3,5,6,7}	

The corresponding set-cover problem would be taken over a base set $\{(u, z)|0 \leq u \neq z \leq 7\}$ and subset family consisting of all the δ_v in the table.

B. SET-COVER $\leq_P$ ID-CODE

We next reduce an identifying code problem to a set cover problem.

Theorem 2 *Given a base set* $\mathbf{U}$ *of* m *elements and a family of subsets* $\mathcal{S}$ *of cardinality* $s = |\mathcal{S}|$, *finding the optimal set cover requires no more computation than finding an identifying code over a directed graph of* $n \leq 2\max(s, m+1) + \lg(m+1)$ *vertices with additional* $O(m+s)$ *operations.*

The following version of this theorem for undirected graphs will be presented in Section V.

Corollary 1 *Finding an optimal set cover with a base set of at least 4 elements, requires no more computation than finding an identifying code over an* undirected *graph of* $2m + \max(s, m+1) \leq n \leq 3m + \max(s, m+1)$ *vertices with additional* $O(ms^2)$ *operations.*

To prove Theorem 2, we first provide and analyze a construction of a specific directed graph (that will be used in the reduction) from an instance of the set cover problem.

Construction 1 *Let* $\mathbf{U} = \{u_1, ..., u_m\}$ *be a base set and* $\mathcal{S} = \{\mathbf{S}_1, ..., \mathbf{S}_{m+1}\}$ *be a family of subsets of* $\mathbf{U}$. *Then we construct a directed graph* $G(V, E)$ *with* $n = 2m + 1$ *vertices* $V = \{v_1, \ldots, v_n\}$ *and edges* E *such that*

1) *The outgoing ball of each vertex* $v_1 \ldots v_{m+1}$ *is constructed to be*

$$B^-(v_i) = \begin{cases} f(\mathbf{S}_i) & \text{if } u_i \in \mathbf{S}_i \\ V - f(\mathbf{S}_i) & \text{otherwise.} \end{cases} \quad (1)$$

where the one-to-one mapping $f : \mathbf{U} \to V$ *is defined by* $f(u_i) = v_i$ *for all* $i \leq m$ *and* $f(\mathbf{S}) = \cup_{u \in \mathbf{S}} f(u)$.

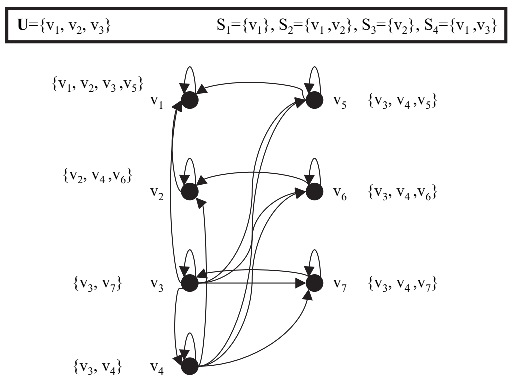


Fig. 2. An example of our reduction framework. Incoming balls are noted near their corresponding vertices.

2) *The outgoing balls of the remaining vertices $v_{m+2} \dots v_n$ are constructed to be:*

$$B^-(v_i) = \{v_i, v_{i-m-1}\}.$$

We next provide several properties of an arbitrary identifying code $\mathbb{C}$ for the graph $G(V,E)$; it might be useful to refer to Figure 2, which demonstrates our construction on a simple example, when reading these properties. Recall that we use the notation $D(v_i, v_j)$ to denote the difference set of the pair $(v_i, v_j) \in V$, and the notation δ_{v_i} to denote the distinguishing set of vertex v_i. We introduce as additional notation the set $\overleftrightarrow{U} = \{(v_i, v_{i+m+1}) \,|\, i \leq m\}$ and corresponding operator $\overleftrightarrow{\delta} = \delta \cap \overleftrightarrow{U}$.

Property 1 *Any identifying code of $G(V,E)$ must contain all vertices $v_{m+2} \dots v_n$.*

Property 2 *For all $k \geq m+2$, the distinguishing set $\overleftrightarrow{\delta_{v_k}}$ is empty.*

Property 3 *$\mathbb{C}$ is an identifying code if and only if $\{v_{m+2} \dots v_n\} \subseteq \mathbb{C}$ and $\{\overleftrightarrow{\delta_{v_i}} \mid i \leq m+1$ and $v_i \in \mathbb{C}\}$ is a cover of $\overleftrightarrow{U}$.*

Proof of Properties 1-3: By construction $B^+(v_{m+1})$ does not contain any vertex of index larger than $m+1$, namely $B^+(v_{m+1}) \cap \{v_{m+2}, ..., v_n\} = \emptyset$. For the rest, *i.e.*, $j \leq m+1$, v_j is in $B^+(v_{m+1})$ if and only if $v_j \in B^+(v_i)$ for all $i \geq m+2$. It follows that the difference sets $D(v_{m+1}, v_i) = \{v_i\}$ for all $i \geq m+2$. To complete the proof of Property 1 we use Lemma 2 that implies that $\{v_i | i \geq m+2\}$ must be contained in any identifying code $\mathbb{C}$.

Property 2 is straightforward.

To prove the forward direction of Property 3, note that Lemma 2 implies that $\overleftrightarrow{\delta_{v \in \mathbb{C}}}$ covers $\overleftrightarrow{U}$. However, Property 2 gives that $\overleftrightarrow{\delta_{v_k}} = \emptyset$ for all $k > m+1$, and what remains, together with Property 1 completes this direction of the proof.

For the converse direction, we show that the latter two conditions in the property statement imply that all difference sets are non-empty, so that Lemma 1 applies to show that $\mathbb{C}$ is an identifying code. We first consider $D_{\mathbb{C}}(v_i, v_j)$ where $(v_i, v_j) \in \overleftrightarrow{U}$; for such pairs in $\overleftrightarrow{U}$, our construction provides that v_i is either in the identifying set $I_{\mathbb{C}}(v_i)$ or else in $I_{\mathbb{C}}(v_j)$, so that $D_{\mathbb{C}}(v_i, v_j) \neq \emptyset$ for all $(v_i, v_j) \in \overleftrightarrow{U}$. For pairs $(v_i, v_j) \notin \overleftrightarrow{U}$ and all $\mathbb{C}$, we observe that

$$D(v_i, v_j) \cap \{v_{m+2}, ..., v_n\} \neq \emptyset, \tag{2}$$

by considering two possibilities for i (assumed $< j$ without loss of generality): (i) $i \leq m$, wherein $I_{\mathbb{C}}(v_i)$ contains v_{i+m+1} and $I_{\mathbb{C}}(v_j)$ cannot; or (ii) $j > m+1$, wherein $I_{\mathbb{C}}(v_j)$ contains v_j and $I_{\mathbb{C}}(v_i)$ cannot. ∎

Note that Property 3 produces a one-to-one correspondence between identifying codes and set covers using distinguishing sets, so that, in fact, a minimum identifying code produces a minimum set cover. We next make use of this property to relate identifying codes to the original subsets $\mathcal{S}$.

The family of distinguishing sets $\{\overleftrightarrow{\delta_{v_i}} | i \leq m+1\}$ over the support $\overleftrightarrow{U}$ is equivalent to the original family of subsets $\mathcal{S}$ over the support $\mathbf{U}$. We use this to develop the following lemma.

Lemma 3 *$\mathbb{C}$ is an identifying code of $G(V,E)$ if and only if $\{v_{m+2} \dots v_n\} \subseteq \mathbb{C}$ and $\{\mathbf{S}_i \mid v_i \in \mathbb{C}, i \leq m+1\}$ is a set cover of $(\mathbf{U}, \mathbf{S})$.*

Proof: Based on Property 3, all we need to show is that there is a one-to-one mapping between the family of distinguishing sets $\{\overleftrightarrow{\delta_{v_i}} | i \leq m+1\}$ over the support $\overleftrightarrow{U}$ and the original family of subsets $\mathcal{S}$ over the support $\mathbf{U}$. In the following the indices i, j are taken to be $i \leq m+1$, and $j \leq m$. By construction if $u_j \notin \mathbf{S}_i$ then vertices v_j and v_{j+m+1} are either both in or both not in $B^-(v_i)$. Otherwise if $u_j \in \mathbf{S}_i$ then only one of them is in $B^-(v_i)$. It follows that $(v_j, v_{j+m+1}) \in \delta_{v_i}$ if and only if $u_j \in \mathbf{S}_i$, completing the proof. ∎

Proof of Theorem 2: Given a base set $\mathbf{U}$ of size m and a family of subsets $\mathcal{S}$ of size s, we trivially produce sets $\mathbf{U}'$ and S' that fit Construction 1 as follows: (i) if $s < m+1$, then $\mathbf{U}'$ is derived from $\mathbf{U}$ by padding it with x new items, where x is the smallest integer satisfying $m + x + 1 \leq s + 2^x$, and S' is derived from S by adding $m+x+1-s$ distinct subsets of these new items (note that $x < 1+\lg m+1$); (ii) otherwise, $\mathbf{U}'$ is derived from $\mathbf{U}$ by padding it with $s-1-m$ new items, and these items are also added to each set in S to form S'. Lemma 3 then assures that a minimum identifying code of the generated graph corresponds to a minimum set cover of $(\mathbf{U}, S)$. ∎

IV. Approximating the optimal identifying code

Given a base set $\mathbf{U}$ of size m and a family of subsets $\mathcal{S}$, there is a well-known greedy approximation of the optimal resulting set cover. This polynomial-time algorithm repeatedly picks (until exhaustion) an unused set in $\mathcal{S}$ that covers the largest number of remaining elements of $\mathbf{U}$. The performance ratio of this algorithm was shown by Slavik [34] to be

$$\frac{s_{\text{greedy}}}{s_{\text{min}}} = \ln m - \ln\ln m + \Theta(1), \tag{3}$$

where s_{min} and s_{greedy} represent the minimum and greedily produced set covers, respectively.

The reduction in Theorem 1 thus provides a straightforward method of translating the greedy set covering algorithm into the greedy identifying code construction in Algorithm 1.

Algorithm 1 Greedy ID-code $\mathtt{ID-GREEDY}(G)$

We start with a graph $G = (V,E)$. The greedy set cover approximation is denoted $\mathtt{SC-GREEDY}(\mathbf{U}, \mathcal{S})$.

```
ID − GREEDY(G) → $\mathbb{C}_{greedy}$
1.      Compute $\{I(u) | u \in V\}$.
2.      Compute $\Delta = \{\delta_u | u \in V\}$.
3.      $\mathbf{C} \leftarrow$ SC − GREEDY$(\mathbf{U}, \Delta)$
4.      Output $\mathbb{C}_{greedy} \leftarrow \{u \in V \mid \delta_u \in \mathbf{C}\}$
```

The remainder of this section is devoted to proving that the reduction in Theorem 1 is tight enough to maintain the approximation guarantees of the set cover solution. This result is formalized with the following theorem.

Theorem 3 *There exists a non-negative constant c_1 such that, for every graph G of n vertices,*

$$\frac{c_{greedy}}{c_{min}} < c_1 \ln n,$$

where c_{min} and c_{greedy} are the sizes of the minimum and greedy identifying codes, respectively.
Furthermore, there exists a non-negative constant $c_2 < c_1$ and a family of graphs for which

$$\frac{c_{greedy}}{c_{min}} > c_2 \ln n.$$

The upper bound of Theorem 3 follows from the fact that the transformation in Theorem 1 maps identifying codes on n vertices to set covers over base sets of size $\frac{n(n-1)}{2}$. As such, since the greedy set cover algorithm has an approximation guarantee of $\frac{s_{\text{greedy}}}{s_{\text{min}}} < c_1' \ln m$, we have that

$$\frac{c_{\text{greedy}}}{c_{\text{min}}} < c_1' \ln \frac{n(n-1)}{2} < 2c_1' \ln n. \tag{4}$$

We will prove the lower bound of the theorem by providing a specific example that attains it in Section IV-B and thereafter.

As a basis for the lower bound example, we first provide some definitions and technical lemmae.

A. Groundwork for the lower bound

The following lemma will be instrumental in our lower bound construction.

Lemma 4 *Consider a collection*[4] *of cardinality m of non-empty sets, $\mathcal{M} = [\mathbf{M}_1, ..., \mathbf{M}_m]$, over a base set $\mathbf{U} = \{u_1, \ldots, u_k\}$ of size $\lg(m) + 2 \leq k \leq m$. Then there is a family of $2m$ different subsets $\mathcal{I} = \{\mathbf{I}_1, ..., \mathbf{I_{2m}}\}$ such that:*

- *$u_i \in \mathbf{I}_i$ for all $i \leq k$, and*
- *$M_i = \mathbf{I}_i \oplus \mathbf{I}_{i+m}$ for all $i \leq m$.*

Proof: Our proof constructively generates $\mathcal{I}$. We start by considering the first set in $\mathbf{M}_1$ and arbitrarily adding to $\mathcal{I}$ a pair of sets $\mathbf{I}_1$ and $\mathbf{I}_{1+m}$ whose symmetric difference is $\mathbf{I}_1 \oplus \mathbf{I}_{1+m} = \mathbf{M}_1$. Note that there are at least $2^{\lg(m)+1} \geq 2m$ such pairs, since the first set in the pair can be any subset of the base set containing u_1, and the second set in the pair would thus be determined uniquely by the desired symmetric difference. As such, we can safely continue to add m distinct pairs of sets in $\mathcal{I}$ with the property that $\mathbf{I}_{2i} \oplus \mathbf{I}_{2i+1} = \mathbf{M}_i$. ∎

For expediency, we shall henceforth assume, without loss of generality, that the elements of $\mathcal{I}$ are arranged so that $\mathbf{I}_i \oplus \mathbf{I}_{i+m} = \mathbf{M}_i$ for all $i \leq m$.

B. Lower bound construction

We now develop the construction that will provide our desired approximation lower bound. Our construction transforms certain instances of the set cover problem into an identifying code problem. The salient point of the construction is that it provides an explicit link between the cardinalities of the minimum (or greedy) set covers in one problem and the minimum (or greedy) identifying codes in the other problem. We shall then make use of an existing result in the literature to show that the desired set cover instances exist. The construction is followed by a detailed example.

Construction 2 *Let $(\mathbf{U} = \{u_1, ...u_m\}, \mathcal{S} = \{\mathbf{S}_1, ...\mathbf{S}_{2m-k}\})$ be a set cover problem. Furthermore, let $\mathcal{S}_{min}$, $\mathcal{S}_{greedy}$, s_{min}, and s_{greedy} be a minimum and greedy set covers and their corresponding cardinalities, and*

[4]We use the term "collection" to represent a multiset, whose contents can have multiplicities. The contents of a collection are enclosed in square brackets, *e.g.,* [1, 1, 3, 2, 2].

assume that $m = 2^k$ and $s_{\mathit{min}} \geq k+2$. Without loss of generality, assume that the subset $\mathcal{S}_{\mathit{min}}$ appears first in $\mathcal{S}$, namely $\mathcal{S}_{\mathit{min}} = \{\mathbf{S}_1, ..., \mathbf{S}_{s_{\mathit{min}}}\}$.

We then generate a graph G from $(\mathbf{U}, \mathcal{S})$ as follows. The graph will have $n = 2m$ vertices, with vertex v_i corresponding to set $\mathbf{S}_i$ for $i \leq m$. To determine the edges of the graph, we shall make use of two collections:

- *$\mathcal{M} = [\mathbf{M}_i]$ is the collection of m sets defined by $\mathbf{M}_i = \{v_j \,|\, \mathbf{S}_j \in \mathcal{S}_{\mathit{min}}$ and $u_i \in \mathbf{S}_j\}$.*
- *$\overline{\mathcal{M}} = [\overline{\mathbf{M}}_i]$ is the collection of m sets such that $\overline{\mathbf{M}}_i = \{v_j \,|\, \mathbf{S}_j \notin \mathcal{S}_{\mathit{min}}$ and $u_i \in \mathbf{S}_j\}$.*

Provided that $k > 1$, Lemma 4 implies the existence of the set:

- *$\mathcal{I} = \{\mathbf{I}_i\}$ having $2m$ distinct sets over the support $\{v_1, ..., v_{s_{\mathit{min}}}\}$ such that:*
  - *$\mathbf{I}_i \oplus \mathbf{I}_{i+m} = \mathbf{M}_i$ for all $i \leq m$, and*
  - *$v_j \in \mathbf{I}_j$ for all j such that $S_j \in \mathcal{S}_{\mathit{min}}$.*

We can also simply generate the following list:

- *$\overline{\mathcal{I}} = [\overline{\mathbf{I}}_i]$ having $2m$ sets over the support $\{v_{s_{\mathit{min}}+1}, ..., v_{2m-k}\}$ such that:*
  - *$\overline{\mathbf{I}}_i \oplus \overline{\mathbf{I}}_{i+m} = \overline{\mathbf{M}}_i$ for all $i \leq m$, and*
  - *$v_j \in \overline{\mathbf{I}}_j$ for all j such that $S_j \notin \mathcal{S}_{\mathit{min}}$.*

This is done by setting $\overline{\mathcal{I}}_i = \emptyset$ and $\overline{\mathcal{I}}_{i+m} = \overline{\mathbf{M}}_i$ for $i \leq m$, and then toggling the existence of u_i and u_{i+m} in sets $\overline{\mathcal{I}}_i$ and $\overline{\mathcal{I}}_{i+m}$ so as to satisfy the stated properties.

The edges of G are then defined in terms of the incoming balls of its vertices:

$$B^+(v_i) = \mathbf{I}_i \cup \overline{\mathbf{I}}_i \cup p_{i-1 \pmod m} \tag{5}$$

where p_i are uniquely chosen subsets over the support $\{v_{2m-k+1} \dots v_{2m}\}$ ensuring that all balls include the vertices themselves, i.e., *$v_j \in B(v_j)$ for $j \geq 2m-k+1$. This can be done by choosing the $(i+1)$-th set in the power set $\mathcal{P}(\{v_{2m-k+1} \dots v_{2m}\})$, where the power set elements are ordered so that the $m-j$-th set contains v_{m-j} for $j \leq k$.*

C. Example of Construction 2

Consider the base set $\mathbf{U} = \{u_1, u_2, u_3, \dots u_8\}$ and the set

$$\begin{aligned}\mathcal{S} \;=\; & \{\{u_1,u_2\},\{u_4,u_3\},\{u_5,u_6\},\{u_1,u_7\},\{u_1,u_8\},\\ & \{u_2,u_3\},\{u_4,u_5\},\{u_1\},\{u_2\},\{u_3\},\\ & \{u_4\},\{u_5\},\{u_6\}\}\end{aligned}$$

In the terminology of Construction 2, we have $k = 3$, $m = 8$, and it is clear that the smallest set covering for $(\mathbf{U}, \mathcal{S})$ is of size $s_{\text{min}} = 5$ because (i) the sets $\{u_1, u_7\}$ and $\{u_1, u_8\}$ are the only sets containing u_7 and u_8 and, thus, must be in a set cover; and (ii) after including these sets, there remain five base elements that must be covered, but no set contains more than two elements (*i.e.,* at least three additional sets are needed). $\mathcal{S}_{\text{min}} = \{\{u_1,u_2\},\{u_4,u_3\},\{u_5,u_6\},\{u_1,u_7\},\{u_1,u_8\}\}$ is an example of an optimal set cover.

We then generate a graph $G = (V, E)$ corresponding to this set cover problem, with $2m$ vertices $v_1 \dots v_{2m}$. Following the construction, we first compute the collections $\mathcal{M}$ and $\overline{\mathcal{M}}$ to be:

$$\begin{aligned}\mathcal{M} \;=\; & [\{v_1,v_4,v_5\},\{v_1\},\{v_2\},\{v_2\},\{v_3\},\{v_3\},\{v_4\},\\ & \{v_5\}]\\ \overline{\mathcal{M}} \;=\; & [\{v_8\},\{v_6,v_9\},\{v_6,v_{10}\},\{v_7,v_{11}\},\{v_7,v_{12}\},\\ & \{v_{13}\},\emptyset,\emptyset].\end{aligned}$$

Intuitively, the i-th set in $\mathcal{M}$ represents the sets that cover the base element u_i in the minimum set cover, whereas the i-th set in $\overline{\mathcal{M}}$ represents the sets that cover u_i but are not in the minimum set cover.

$$
M(\mathbf{U},\mathcal{S}) = \left[\begin{array}{c||ccccc|cccccccc|ccc}
 & \multicolumn{5}{c|}{\mathcal{I}} & \multicolumn{8}{c|}{\overline{\mathcal{I}}} & \multicolumn{3}{c}{\mathcal{P}} \\
 & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 & v_8 & v_9 & v_{10} & v_{11} & v_{12} & v_{13} & v_{14} & v_{15} & v_{16} \\
\hline\hline
v_1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
v_2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\
v_3 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
v_4 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\
v_5 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
v_6 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
v_7 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
v_8 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
\hline
v_9 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
v_{10} & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\
v_{11} & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\
v_{12} & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
v_{13} & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\
v_{14} & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\
v_{15} & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
v_{16} & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1
\end{array}\right]
$$

Fig. 3. The adjacency matrix of a graph $G(\mathbf{U},\mathcal{S})$ from Construction 2.

Utilizing Lemma 4, we also construct the set $\mathcal{I}$ and the collection $\overline{\mathcal{I}}$:

$$
\begin{aligned}
\mathcal{I} \;=\{\; & \{v_1\},\{v_2\},\{v_2,v_3\},\{v_2,v_4\},\{v_3,v_5\},\\
& \{v_1,v_3,v_4\},\{v_1,v_4,v_5\},\{v_1,v_2,v_3,v_5\},\\
& \{v_4,v_5\},\{v_1,v_2\},\{v_3\},\{v_4\},\{v_5\},\\
& \{v_1,v_4\},\{v_1,v_5\},\{v_1,v_2,v_3\}\;\}\\
\overline{\mathcal{I}} \;=[\; & \{v_9\},\{v_{10}\},\{v_{11}\},\{v_{12}\},\{v_{13}\},\{v_6\},\{v_7\},\\
& \{v_8\},\{v_8,v_9\},\{v_6,v_9,v_{10}\},\{v_6,v_{10},v_{11}\},\\
& \{v_7,v_{11},v_{12}\},\{v_7,v_{12},v_{13}\},\{v_6,v_{13}\},\{v_7\},\\
& \{v_8\}\,].
\end{aligned}
$$

Note that $\mathcal{I}$ and $\overline{\mathcal{I}}$ are not unique, in the sense that there is a variety of such collections that are consistent with the construction.

Finally, applying (5) with the power set

$$
\begin{aligned}
\mathcal{P}\left(\{v_{14},v_{15},v_{16}\}\right) &= \{p_0,\ldots,p_7\}\\
&= \{\emptyset,\{v_{16}\},\{v_{15}\},\{v_{15},v_{16}\},\\
&\quad \{v_{14}\},\{v_{14},v_{16}\},\{v_{14},v_{15}\},\\
&\quad \{v_{14},v_{15},v_{16}\}\}
\end{aligned}
$$

provides the edges of the graph in terms of incoming balls of vertices, the first few of which are:

$$
\begin{aligned}
B^{+}(v_1) &= \{v_1,v_9\}\\
B^{+}(v_2) &= \{v_2,v_{10},v_{16}\}\\
B^{+}(v_3) &= \{v_2,v_3,v_{11},v_{15}\}
\end{aligned}
$$

It is easier to conceptualize the graph in terms of its adjacency matrix, as depicted in Figure 3. In this matrix, each row represents the incoming ball of a vertex. We shall prove with Properties 1- 3 that $c_{\text{min}} = s_{\text{min}}$ and $c_{\text{greedy}} = s_{\text{greedy}} + k$ for this graph, where c_{min} and c_{greedy} are the minimum and greedy identifying code cardinalities for $G(\mathbf{U}, \mathcal{S})$.

D. Lower bound

We next provide some properties of Construction 2 that will be crucial in completing the proof of the lower approximation bound of Theorem 3. Here $\overleftrightarrow{U} = \{(v_i, v_{m+i}) | i \le m\}$, and recall that $\overleftrightarrow{\delta} = \delta \cap \overleftrightarrow{U}$.

Property 4 *Given a set cover problem $(\mathbf{U}, \mathcal{S})$ with $|\mathbf{U}| = 2^k = m$, $|\mathcal{S}| = 2m - k$, and $s_{\mathit{min}} \ge k + 2$, Construction 2 produces a graph G with the following properties:*

1) *The vertices $V_{\mathit{min}} = \{v_1, ..., v_{s_{\mathit{min}}}\}$ associated with $\mathcal{S}_{\mathit{min}}$ form an identifying code of $G = (V, E)$. More precisely, V_{min} contains exactly vertices v_i, where i is such that $S_i \in \mathcal{S}_{\mathit{min}}$.*
2) *The distinguishing sets of $\hat{V} = \{v_{2m-k+1}, ..., v_{2m}\} \subseteq V$ cover all pairs of distinct vertices except $\overleftrightarrow{U}$. More precisely,*
$$\bigcup_{i \in \{2m-k+1 \ldots 2m\}} \delta_{v_i} = \{(v_u, v_z) \,|\, z \ne u + m \textit{ and } 1 \le u < z \le 2m\}.$$
3) *The modified set cover problem $(\overleftrightarrow{U}, \{\overleftrightarrow{\delta_{v_1}} \ldots \overleftrightarrow{\delta_{v_{2m-k}}}\})$ is equivalent to the original problem $(\mathbf{U}, \mathcal{S})$. As such, the function $f : \overleftrightarrow{U} \longrightarrow \mathbf{U}$ where $f\left((v_i, v_{i+m})\right) = u_i$ has the property that*
$$f(\overleftrightarrow{\delta_{v_i}}) = S_i,$$
with the usual understanding that $f(S) = \cup_{s \in S} f(s)$. Note that this also provides an equivalence between covers in the modified problem and covers in the original problem.

It may be beneficial to refer to Figure 3 while reading the proof.

Proof: The first property follows from the fact that, by design, the sets in $\mathcal{I}$ are all different, meaning that the symmetric difference of $B^+(v_i) \cap V_{\text{min}}$ and $B^+(v_j) \cap V_{\text{min}}$ is non-empty for all distinct vertices v_i and v_j. Lemma 1 thus implies that V_{min} is an identifying code.

To prove the second property, note that, by construction, $B^+(v_i) \cap \hat{V}$ is unique for every $1 \le i \le m$, and similarly for every $m+1 \le i \le 2m$. In fact, only for $j = i + m$ is $B^+(v_i) \cap \hat{V} = B^+(v_j) \cap \hat{V}$, hence proving the property.

To prove the third property, we note that, by definition, $(v_j, v_{j+m}) \in \overleftrightarrow{\delta_{v_i}}$ means that $v_i \in D(v_j, v_{j+m})$. By construction the supports of $\mathcal{I}$ and $\overline{\mathcal{I}}$ are disjoint, and their union is $\{v_1, \ldots, v_{2m-k}\}$. Furthermore, by the construction of $\mathcal{I}$ (and $\overline{\mathcal{I}}$), $(v_j, v_{j+m}) \in \overleftrightarrow{\delta_{v_i}}$ if and only if $v_i \in \mathbf{M}_j$ (and $\overline{\mathbf{M}}_j$) where $j \le m$ and v_i is taken over the support of $\mathcal{I}$ ($\overline{\mathcal{I}}$). Therefore by the definition of $\mathcal{M}$ and $\overline{\mathcal{M}}$ it follows that $(v_j, v_{j+m}) \in \overleftrightarrow{\delta_{v_i}}$ if and only if $u_j \in S_i$, completing the proof. Note that this property can be extended to the set of all distinguishing sets, *i.e.*, $\{\overleftrightarrow{\delta_{v_1}} \ldots \overleftrightarrow{\delta_{v_{2m-k}}}\}$, if we allow padding $\mathcal{S}$ of the original set cover problem with k empty subsets. ∎

For the next property we need some additional definitions. A code C is said to be a partial code if $C \subset \mathbb{C}$ is not an identifying code. Any partial code C partitions the set of vertices into indistinguishable subsets, $\{\mathit{INDIST}_i(C)\}$, where all $v \in \mathit{INDIST}_i(C)$ have an identical identifying set.

We also need to make a *consistency assumption* in the implementation of $\mathtt{SC-GREEDY}$, the greedy set covering algorithm. Specifically, recall that $\mathtt{ID-GREEDY}$ calls $\mathtt{SC-GREEDY}$ with a base set $\{(v_i, v_j) | i \ne j\}$ and a family of subsets $\{\delta_u\}$. Our assumption, without loss of generality, will thus be that when $\mathtt{SC-GREEDY}$ must choose between subsets δ_u and δ_v, both covering an equal number of uncovered base elements, it will break ties in favor of the first vertex to appear in the precedence list $[v_{2m-k+1}, \ldots, v_{2m}, v_1, \ldots, v_{2m-k}]$.

Property 5 *Vertices $\{v_{2m-k+1}, ..., v_{2m}\}$ are guaranteed to appear in the greedy identifying code, if* $\texttt{SC}-\texttt{GREEDY}$ *(in* $\texttt{ID}-\texttt{GREEDY}$*) breaks ties in favor of the distinguishing sets that correspond to these vertices.*

Proof: In each iteration of $\texttt{SC}-\texttt{GREEDY}$ (as part of $\texttt{ID}-\texttt{GREEDY}$) a codeword is added to a partial code C, which partitions the set of vertices into indistinguishable subsets $\{\mathit{INDIST}_i\}$. A vertex $v \notin C$ distinguishes exactly $\sum_i |\mathit{INDIST}_i \cap B^-(v)||\mathit{INDIST}_i \setminus B^-(v)|$ undistinguished pairs so far. Clearly this expression is maximized when $|\mathit{INDIST}_i \cap B^-(v)| = |\mathit{INDIST}_i \setminus B^-(v)|$ for all i, meaning that the added codeword partitions each indistinguishable set of vertices into two equally-sized indistinguishable subsets (*i.e.,* one subset containing the codeword in their identifying set, and the other not). In fact, the partial code $C = \{v_{2m-k+1}, \ldots, v_{2m}\}$ exhibits this best-case behavior, in that its vertices, considered one after another, always split the existing indistinguishable sets exactly in half. As such, assuming ties are broken in favor of these vertices, $\texttt{SC}-\texttt{GREEDY}$ will always choose the vertices of C before any other vertex in our constructed graph.

∎

The following corollary is a direct consequence of Properties 4 and 5.

Corollary 2 *The directed graph G generated by Construction 2 has the following properties:*

1) $c_{min} = s_{min}$, *and*
2) $c_{greedy} = s_{greedy} + k$.

Proof: Property 4.1 implies that $c_{\text{min}} \leq s_{\text{min}}$. To prove the other direction by Lemma 1 the distinguishing sets $\overleftrightarrow{\delta}_{v\in\mathbb{C}}$ should cover $\overleftrightarrow{U}$ for any identifying code, $\mathbb{C}$. Since by Property 4.3 this set cover problem is equivalent to the original one, it implies that $c_{\text{min}} \geq s_{\text{min}}$, thus proving that $c_{\text{min}} = s_{\text{min}}$.

Under our consistency assumption $\texttt{SC}-\texttt{GREEDY}$ breaks ties in favor of the first vertex to appear in the precedence list $[v_{2m-k+1}, \ldots, v_{2m},\ v_1, \ldots, v_{2m-k}]$. In this way, we are guaranteed (by Property 5) that $\texttt{ID}-\texttt{GREEDY}$ will first select vertices $v_{2m-k+1} \ldots v_{2m}$, which under Property 4.2 will leave only $|\overleftrightarrow{U}| = m$ pairs to be covered, and then by Property 4.3 it will select vertices corresponding to a greedy set cover over the original problem, giving a total of $s_{\text{greedy}} + k$ vertices. ∎

We may now conclude the main proof of this section.

Proof of lower bound of Theorem 3: Slavik [34] demonstrated that there exist set cover problems $(\mathbf{U}, \mathcal{S})$ with greedy covers of any size s_{greedy} and minimum covers of size $s_{\text{min}} \geq 2$ as long as the size $|\mathbf{U}| = m$ can be lower bounded by the function $N(s_{\text{greedy}}, s_{\text{min}})$, where

$$\begin{aligned} &\ln N(k', l) \leq \\ &\quad \ln l + \tfrac{2l-1}{2l(l-1)} \left[(k'-l) + (l-2) \left(1 - \left(\tfrac{l-1}{l}\right)^{k'-l} \right) \right], \\ &\quad \text{for any } k' \geq l \geq 2, \end{aligned} \tag{6}$$

which can be (weakened and) simplified to $\ln l + \frac{k'}{l-1}$. As such, we can see that, for $s_{\text{greedy}} \geq s_{\text{min}} \geq 2$, if

$$\ln m \geq \ln s_{\text{min}} + \frac{s_{\text{greedy}}}{s_{\text{min}} - 1}, \tag{7}$$

then $m > N(s_{\text{greedy}}, s_{\text{min}})$, and a corresponding set cover problem exists.

In order to apply Construction 2 to a set cover problem $(\mathbf{U}, \mathcal{S})$ produced by Slavik's construction, we need to ensure that the construction's assumptions are satisfied, namely that (i) $m = 2^k$, (ii) $s_{\text{min}} = k+2$, and that (iii) $|\mathcal{S}| = 2m - k$. In addition, we need the constructed graph to have the property that $\frac{c_{\text{greedy}}}{c_{\text{min}}} \geq$

$c \ln n = c \ln m + c \ln 2$ for some constant c in order to have our performance bound; under Corollary 2, this corresponds to a condition (iv) that $\frac{s_{\text{greedy}}+k}{s_{\text{min}}} = c \ln m + c \ln 2$, which reduces to

$$s_{\text{greedy}} = ck^2 + 2ck + 2c \ln 2. \tag{8}$$

Thus, if we have set cover problems satisfying conditions (i)-(iv), then we can create an identifying code instance satisfying the lower bound of Theorem 3.

Slavik's construction trivially satisfies conditions (i) and (ii) since (7) holds for any given $m, s_{\text{min}} \geq 2$. Reconciling (7) with (8), we see that condition (iv) is satisfied when $ck^2 + 2ck + 2c \ln 2 \leq k^2 + (\ln 2 - \ln(k+2))k - \ln(k+2)$, which will clearly hold for any $0 \leq c < 1$ when k is sufficiently large. Finally, we can transform a set cover problem $(\mathbf{U}, \mathcal{S})$ satisfying conditions (i),(ii), and (iv) into a problem $(\mathbf{U}, \mathcal{S}')$ satisfying (i)-(iv) as follows:

If $|\mathcal{S}| < 2m - k$, then we can pad $\mathcal{S}$ with empty sets to get $\mathcal{S}'$ of cardinality $2m - k$ without breaking Construction 2.

If $|\mathcal{S}| > 2m - k$, then we can take $\mathcal{S}' = \mathcal{S}_{\text{min}} \cup S_{\text{greedy}}$ without violating any conditions. This new set will have cardinality $|\mathcal{S}'| \leq s_{\text{min}} + s_{\text{greedy}} \leq (k+2) + ck^2 + 2ck + 2c \ln 2$, which is clearly $\leq m$ for $c = 1$ and large k. ∎

V. Hardness of Approximation

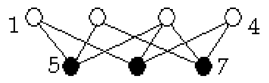
We next manipulate the reductions of Theorem 2 to carry set cover hardness results over to the identifying code problem. Our work is based on the work of Feige [28] proving that (for any $\epsilon > 0$) no polynomial-time approximation of set cover can attain a performance ratio of $(1 - \epsilon) \ln m$ unless NP$\subset$DTIME($m^{\lg \lg m}$). Our proof differs from the more general hardness result of [38] for the test cover problem because of the constraints imposed by the (undirected) graph structure on which identifying codes are defined (rather than the arbitrary "tests" permitted in the test cover problem).

Theorem 4 *For any $\epsilon > 0$ the identifying code problem has no polynomial time approximation with performance ratio $(1 - \epsilon) \ln n$ (for directed and undirected graphs) unless NP$\subset$DTIME($n^{\lg \lg n}$).*

In order to prove the hardness result for undirected graphs (and hence also for directed graphs) we first provide a proof of Corollary 1 for the undirected graph case. Recall that Corollary 1 states that solving a set-cover problem is computationally equivalent to finding the optimal identifying code over undirected graphs. We will then extend the proof to carry over the set cover hardness result of Feige [28]. Other hardness results with looser complexity assumptions can also be applied to get slightly weaker results. For example, the work in [30] can be applied under the assumption P $\neq$ NP) to show inapproximability within a factor of $c \ln n$ for some smaller constant c.

A. Notation

For convenience of exposition, we shall represent an undirected graph by a binary symmetric matrix G. In this matrix, rows or columns are *characteristic vectors* of corresponding *balls*, in that their i-th entries are 1 if and only if the i-th vertices are in the corresponding ball. Note that this matrix coincides with the adjacency matrix of the graph if all its vertices have self loops. Therefore we will refer to G as the *adjacency* matrix, bearing in mind that its diagonal is all ones. Selecting codewords for an identifying code can thus be viewed as selecting columns to form a matrix of size $n \times |C|$. We will refer to this matrix as the *code* matrix. It is easy to see that the rows of the code matrix are the characteristic vectors of the corresponding identifying sets. A code is thus identifying if all the rows in the code matrix are unique, or equivalently, if the *Hamming distance* between every two rows in the code matrix is at least one. Note that this definition of identifying code differs from the traditional one by allowing the all zero



$$G = \left[\begin{array}{c|c} 100 & \\ 010 & B_s \\ 001 & \\ \hline 000 & \\ 111 & \\ 101 & B_m \\ 011 & \end{array}\right], \qquad \begin{array}{rl} & \quad\;\; C_7 \\ B_s = & \left[\begin{array}{cc} 0 & 110 \\ 0 & 101 \\ 0 & 111 \end{array}\right] \\ B_m = & \left[\begin{array}{cc} 1 & 011 \\ 0 & 100 \\ 1 & 010 \\ 1 & 001 \end{array}\right] \end{array}$$

Fig. 4. An example of the construction of B with $m = 4,\ s = 3$. The code C_{m+s} corresponds to the solid vertices indexed 5,6 and 7.

row in the code matrix. We say that a column *distinguishes or covers* a pair of rows if it contributes to their Hamming distance. Hence a code is identifying if the corresponding set of columns covers all possible pairs of rows.

For example consider the graph in Fig. 4 and its adjacency matrix G. The last 3 columns denoted by C_7 represent 3 vertices that constitute a code matrix. Since all the rows of C_7 are unique sequences, and hence the minimum Hamming distance is at least 1, the code is guaranteed to be identifying. Note that C_7 corresponds to the minimum identifying code, since at least $3 = \lceil \lg 7 \rceil$ columns are required to represent 7 different binary sequences. The role of m and s in the figure will become clear in the next section.

We shall also use matrix notation to represent the set cover problems. Specifically, we can translate a set cover problem $(\mathbf{U}, \mathcal{S})$ of m base elements into a binary matrix $A \in \{0,1\}^{m \times s}$ whose columns are the characteristic vectors of the s subsets in the set $\mathcal{S}$. As such, solving the set cover problem trivially corresponds to finding a minimum set of columns of A with the property that every row has at least one non-zero element (see Fig. 5). In this formulation, each row of the matrix corresponds to an element in the set cover problem, and we say that a column *covers* a row if their intersecting entry of the matrix contains a 1.

$$\mathcal{S} = \left\{ \begin{array}{c} \{1,2,3,5\}, \\ \{1,3,4,6\}, \\ \{2,3,4,7\} \end{array} \right\} \qquad \Leftrightarrow \qquad A = \begin{pmatrix} 110 \\ 101 \\ 111 \\ 011 \\ 100 \\ 010 \\ 001 \end{pmatrix}$$

Fig. 5. Example of matrix notation, A, of a set cover problem with the base set $\{1, 2, .., 7\}$ and a family of subsets, $\mathcal{S}$.

B. Proof of Corollary 1 for undirected graphs

The proof is based on a construction of an undirected graph G (out of a given arbitrary set cover problem, A), whose minimum identifying code has a one-to-one mapping to a minimum set cover of the given problem.

In the construction of G we use the well known fact [27] that an undirected graph of n vertices can be constructed with an identifying code of size $\lceil \lg n \rceil$ (assuming that an empty set is a valid identifying set). Let $C_n \in \{0,1\}^{n \times \lceil \lg n \rceil}$ denote the columns that correspond to such a code, and let $B \in \{0,1\}^{m+s \times m} \supseteq C_{m+s}$ be the m last columns of an $m+s$ adjacency matrix of an undirected graph possessing a size-$\lg(m+s)$ identifying code C_{m+s} (see Figure 4). Note that we assume here that $m \geq \lg(m+s)$, an assumption that will be justified later. We further divide B into two matrices, B_s and B_m, which include the first s and last m rows of B, respectively. Note that B_m is symmetric by definition.

The first step in the construction is to tie the arbitrary dimensions of the set cover problem A (*i.e.,* the number of base elements m and the number of subsets s) while keeping the minimum set cover unchanged. Toward this end we say that set cover problems $(\mathbf{U}, \tilde{\mathcal{S}})$ and $(\mathbf{U}, \mathcal{S})$ are *equivalent* if $\mathcal{S} \subseteq \tilde{\mathcal{S}}$, and if no minimum set cover $\tilde{SC}$ of $(\mathbf{U}, \tilde{\mathcal{S}})$ includes subsets from $\tilde{\mathcal{S}} - \mathcal{S}$. Equivalently we can say that equivalent set cover problems have identical minimum set covers.

Given an arbitrary set cover problem A, we generate an equivalent set cover problem, $\tilde{A}$, by adding empty subsets, or equivalently by adding zero columns to A. We denote the number of columns of $\tilde{A}$ by $\tilde{s}$, and we use the notation $\mathbf{c}^{m\times n}$ to denote an $m \times n$ matrix whose entries are all c.

Construction 3 *Given a set cover problem $A \in \{0,1\}^{m\times s}$ with $s < 2^m - 2m$, let $s \le \tilde{s} \le 2^m - m$. Generate the binary matrices $\tilde{A} = \begin{pmatrix} A & \mathbf{0}^{m\times(\tilde{s}-s)} \end{pmatrix}$, $B_{\tilde{s}}$, and B_m, and construct the following $n = 2m + \tilde{s}$ undirected graph with adjacency matrix G:*

$$G = \begin{pmatrix} \mathbf{1}^{\tilde{s}\times\tilde{s}} & \tilde{A}^T + B_{\tilde{s}} & B_{\tilde{s}} \\ \tilde{A} + B_{\tilde{s}}^T & B_m & B_m \\ B_{\tilde{s}}^T & B_m & B_m \end{pmatrix},$$

where summation is over $GF(2)$ and a superscript T denotes the transpose operation.

We observe that the matrix G is a valid representation of an undirected graph for our purposes, as it is symmetric with ones along its main diagonal, and we index the columns and rows of G from 1 up to $2m + \tilde{s}$ and label corresponding vertices with $v_1, ..., v_{2m+\tilde{s}}$.

The following are some basic properties of this construction. Recall that we say that a column *covers* a pair of rows if it contributes to their Hamming distance.

Property 6 *Only columns $1, ..., s$ can cover the pairs of rows indexed $(i, i+m)$, for $i = \tilde{s}+1, ..., \tilde{s}+m$. Moreover, any set of columns, SC, covers these pairs of rows if and only if the same set of columns is a set cover of the original problem.*

Property 7 *The $\lceil \lg(m+\tilde{s}) \rceil$ columns of $C_{m+\tilde{s}}$ that are contained in B can cover all the pairs of rows but the ones indexed $(i, i+m)$, for $i = \tilde{s}+1, ..., \tilde{s}+m$.*

Property 8 *Only columns $\tilde{s}+1, ..., 2m+\tilde{s}$ can distinguish between the mixed pairs of rows $1, ..., \tilde{s}$.*

Proof of Properties 6-8: Observe that the bitwise XOR of all row pairs $(i, i+m)$, for $i = \tilde{s}+1, ..., \tilde{s}+m$, results in a row that contains the i-th row of matrix $\tilde{A}$ followed by $2m$ zeros. If we stuck these m rows together they will form the block matrix $(\tilde{A} \;\; \mathbf{0}^{m\times 2m}) = (A \;\; \mathbf{0}^{m\times 2m+\tilde{s}-s})$. Hence any identifying code, which by definition must cover all pairs of rows and in particular all $(i, i+m)$ pairs, must contain a nonempty subset of the first s columns, SC. Moreover, since $(A \;\; \mathbf{0}^{m\times 2m+\tilde{s}-s})$ and A are equivalent it follows that SC is also a valid set cover of the original problem A. Property 6 follows trivially from this observation.

Similarly, Property 8 follows trivially from observing the bitwise XOR of the mixed pairs of rows $1, ..., \tilde{s}$. All the $\binom{\tilde{s}}{2}$ resulting row vectors will have zeros in their first $\tilde{s}$ elements. Hence the first $\tilde{s}$ columns (and hence SC) cannot cover these pairs of rows. To see why Property 7 is true, recall that $C_{m+\tilde{s}}$ corresponds to an identifying code of a graph which contains B. As such, it must cover all mixed pairs of rows of B. The property follows from the observation that all pairs of rows but $(i, i+m)$, for $i = \tilde{s}+1, ..., \tilde{s}+m$, contain some mixed pair of rows of B. ∎

Let $\mathbb{C}$, $\mathcal{S}_{\text{min}}$, c_{min} and s_{min} denote an optimum identifying code for the graph G and a minimum set cover of the original problem A and their sizes, respectively.

Lemma 5

$$s_{min} + \lceil \lg \tilde{s} \rceil \le c_{min} \le s_{min} + \lceil \lg(m+\tilde{s}) \rceil$$

Proof: Given a set cover problem A satisfying the conditions of Construction 3, let $\mathcal{S}_{\text{min}}$ be its minimum set cover of cardinality s_{min}, and let G be the graph of Construction 3. By Lemma 2 every identifying code of G must cover the difference sets of all the pairs of distinct vertices, $U = \{(u, z) \in V^2 \,|\, u \neq z\}$. Partition U into the pairs indexed $U_1 = \{(i, i+m)|\; i = \tilde{s}+1, ..., \tilde{s}+m\}$, and the rest $U \setminus U_1$. Property 6 asserts that any cover of the pairs U_1 is also a cover of the original set cover problem A. Therefore (i) a feasible cover of U_1 is the set of columns (denoted by SC) that corresponds to $\mathcal{S}_{\text{min}}$, and furthermore (ii) due to the optimality of $\mathcal{S}_{\text{min}}$ there is no smaller set of columns that can cover U_1.

To prove the upper bound combine (i) above with Property 7 to generate a feasible identifying code by the set of vertices corresponding to the union of $C_{m+\tilde{s}}$ and SC.

To prove the lower bound, we observe that by property 8 SC cannot cover any of the mixed pairs of rows indexed $1, ..., \tilde{s}$, and therefore it cannot be an identifying code by itself. By (ii) above and since we need at least $\lceil \lg \tilde{s} \rceil$ columns to cover all of the mixed pairs of rows indexed $1, ..., \tilde{s}$, the lower bound follows. ∎

Proof of Corollary 1: Given a set cover problem $(\mathbf{U}, \mathcal{S})$ with $m \geq 4$, and an algorithm for solving the identifying codes problem over undirected graphs, $\texttt{ID}-\texttt{UNDIR} : G \rightarrow \mathbb{C}$, perform the following:

1) Remove *redundant* subsets in $\mathcal{S}$ to produce $\mathcal{S}'$ and construct matrix A out of it. A subset $S_i \in \mathcal{S}$ is redundant if there exist $j \neq i$ s.t. $S_j \in \mathcal{S}$, and $S_i \subseteq S_j$.
2) Construct the undirected graph G according to Construction 3 with $\tilde{s}$ that satisfies $\lceil \lg(m + \tilde{s}) \rceil = \lceil \lg \tilde{s} \rceil$.
3) $\mathbb{C} \leftarrow \texttt{ID}-\texttt{UNDIR}(G)$
4) Output $\{S_i | v_i \in \mathbb{C}, 1 \leq i \leq s\}$

We first prove correctness. Step 1, which can be completed by $O(ms^2)$ operations, is necessary to make sure that the condition $s < 2^m - 2m$ in Construction 3 is satisfied. It is obvious that the set cover problems $(\mathbf{U}, \mathcal{S})$ and $(\mathbf{U}, \mathcal{S}')$ are equivalent (every removed subsets that may be in $\mathcal{S}_{\text{min}}$ of the original problem can be replaced by the subset that contains it while keeping $\mathcal{S}_{\text{min}}$ a valid set cover). The largest size of such un-redundant $\mathcal{S}'$ is $\binom{m}{\lceil \frac{m}{2} \rceil} < \frac{\sqrt{2} 2^m}{\pi \sqrt{m}}$, and it can be verified numerically that $m \geq \lg(2m + |\mathcal{S}'|)$ is satisfied for $m \geq 4$. Next, since the imposed requirement $\lceil \lg(m + \tilde{s}) \rceil = \lceil \lg \tilde{s} \rceil$ can be satisfied with

$$m + 1 \leq \tilde{s} \leq s + m$$

the second condition in Construction 3, *i.e.*, $\tilde{s} \leq 2^m - m$, is met. Finally, Property 6 assures that the output of the reduction is a set cover of $(\mathbf{U}, \mathcal{S})$.

To see why it is a minimum set cover, we use Lemma 5 and the fact that in our construction $\lceil \lg(m + \tilde{s}) \rceil = \lceil \lg \tilde{s} \rceil$ (and hence SC is of the size of a minimum set cover of the original problem).

To complete the proof we observe that the size of the graph G, $n = 2m + \tilde{s}$, is bounded by $2m + \max(s, m+1) \leq n \leq 3m + \max(s, m+1)$.

∎

Construction 3 together with Lemma 5 can serve as a basis for a hardness result by generating $\tilde{s}$ disjoint copies of the original set cover problem before plugging it into Construction 3. Particularly, Raz and Safra's result [30] can be applied in this manner to show that no polynomial-time approximation that performs within a factor of $O(\log n)$ from an optimal identifying code exists, unless P=NP. A similar approach is taken in the next section, which uses an explicit construction, and the hardness result of Feige [28] to yield a tighter result.

C. Proof of Theorem 4

We next prove Theorem 4 for the undirected graph case (and hence for directed graphs as well). We extend Construction 3 to k disjoint copies of the original set cover problem. This construction is then used to carry over the set cover hardness result of Feige [28], *i.e.*, for any $\epsilon > 0$ there is no polynomial

$$G^k = \left[\begin{array}{cccc|cccc|c}
I & I+\tilde{A}^T & \cdots & I+\tilde{A}^T & \tilde{A}^T+B_{\tilde{s}} & B_{\tilde{s}}^T & \cdots & B_{\tilde{s}}^T & B_{\tilde{s}} \\
I+\tilde{A} & I & \ddots & \vdots & B_{\tilde{s}} & \tilde{A}^T+B_{\tilde{s}} & \ddots & \vdots & B_{\tilde{s}} \\
\vdots & \ddots & \ddots & I+\tilde{A}^T & \vdots & \ddots & \ddots & B_{\tilde{s}}^T & \vdots \\
I+\tilde{A} & \cdots & I+\tilde{A} & I & B_{\tilde{s}}^T & \cdots & B_{\tilde{s}}^T & \tilde{A}^T+B_{\tilde{s}} & B_{\tilde{s}} \\
\hline
\tilde{A}+B_{\tilde{s}}^T & B_{\tilde{s}}^T & \cdots & B_{\tilde{s}}^T & B_m & B_m & \cdots & B_m & B_m \\
B_{\tilde{s}} & \tilde{A}+B_{\tilde{s}}^T & \ddots & \vdots & B_m & B_m & & B_m & B_m \\
\vdots & \ddots & \ddots & B_{\tilde{s}}^T & \vdots & & \ddots & \vdots & \vdots \\
B_{\tilde{s}} & \cdots & B_{\tilde{s}} & \tilde{A}+B_2^T & B_m & B_m & \cdots & B_m & B_m \\
\hline
B_{\tilde{s}}^T & B_{\tilde{s}}^T & \cdots & B_{\tilde{s}}^T & B_m & B_m & \cdots & B_m & B_m
\end{array}\right]
\begin{array}{l}
red_1 \\ red_2 \\ \vdots \\ red_k \\ \hline blue_1 \\ blue_2 \\ \vdots \\ blue_k \\ \hline white
\end{array}$$

Fig. 6. The adjacency matrix of the graph G^k in Construction 4.

time algorithm that can approximate the minimum set cover within the performance ratio $(1-\epsilon)\ln m$ unless NP$\subset$DTIME$(m^{\lg\lg m})$.

Our construction requires that the number of subsets s is at most m. This assumption is valid as the set-cover hardness proof in [28] uses a reduction to MAX 3SAT-5 problem from a set cover problem over a family of problems called a *partition system*. In this family there are m elements in the base set, $\mathbf{U}$, and a set $\mathcal{S}$ of subsets each of size m/k, with k being a large constant. Each subset is associated with $k-1$ other pairwise disjoint subsets of size m/k that together partition $\mathbf{U}$ into k equal parts. A "good" cover of $\mathbf{U}$ by disjoint subsets requires only k subsets. A "bad" cover needs roughly d subsets, not belonging to the same partition where $(1-1/k)^d \simeq 1/m$. As k grows, d tends to $k\ln m$, resulting in the desired ratio of $\ln m$. The partition system consists of L partitions, resulting in a total of $s = Lk$ subsets. To construct the partition system Feige used a deterministic construction by Naor et al. [41] with the following parameters, k is an arbitrary large constant, $L = 2^l$, $m = (5n)^{\frac{2l}{\epsilon}}$, with $\epsilon \ll 1$ and where n is the number of variables in the MAX 3SAT-5 problem, and $l = \Theta(\log\log n)$. Clearly the ratio of the number of subsets to base elements scales as $\frac{s}{m} = n^{-O(l)} < 1$.

In the context of our construction we henceforth assume without loss of generality that $s \leq m$.

Let I be the $m\times m$ identity matrix and recall that $B \in \{0,1\}^{m+s\times m}$ is the m last columns of an $m+s$ adjacency matrix, which also include C_{m+s}, and that we further divide B into B_s and B_m, respectively the first s and last m rows of B.

Construction 4 *Given a set-cover problem $A \in \{0,1\}^{m\times s}$, with $s \leq m$, we generate an undirected graph, G^k, of size $n = (2k+1)m$ for some positive integer k as follows.*

1) *Pad A with $m-s$ zero columns to get the matrix $\tilde{A}$ with $\tilde{s} = m$, and generate $B_{\tilde{s}}$ and B_m, all square matrices of size $m\times m$.*
2) *Generate an undirected graph with the adjacency matrix G^K as shown in Figure 6.*

We first observe that G^k is a valid adjacency matrix of an undirected graph.

To make our next observations clearer we divide the rows of the adjacency matrix G^k into 3 blocks which we denote by colors red, $blue$ and $white$. red rows are the first km rows, $blue$ rows are km rows $km+1,...,2km$. and the white rows are the last m rows. We further divide the red and $blue$ rows into k sub blocks of m rows each, red_i, $blue_i$, for $i = 1,...,k$, and the j-th row in a sub block is denoted by $red_i(j)$.

We denote by $(red_i, blue_j)$ the set of m pairs $\{(red_i(1), blue_j(1)),...,(red_i(m), blue_j(m))\}$.

We divide the columns of G^k into blocks of m columns, col_i, for $i = 1,...,2k+1$. Recall that col_{2k+1} include $1+\lceil\lg m\rceil$ columns, C_{2m}, which form an identifying code for a graph of size $2m$.

The proof of the following properties is similar to that of Properties 6, 7 and is omitted.

Property 9 *Only columns col_i can cover the pairs of rows $(blue_i, white)$, for $i = 1,...,k$. Moreover, any set of columns, $SC_i \subseteq col_i$, covers the pairs $(blue_i, white)$ if and only if SC_i is a set cover of the original problem (taking the indices of the columns modulo m).*

Property 10 *The set of pairs (red_i, red_j) and $(blue_i, blue_j)$, for all $i < j$, are covered by $\bigcup_{l=1}^{k} SC_l$.*

Property 11 *All the pairs other than the ones mentioned in Properties 9,10 are covered by a subset of columns in col_{2k+1}, which corresponds to C_{2m}.*

Let c_{min}, s_{min} be the sizes of the minimum identifying code for G^k and the minimum set cover of the original problem.

Lemma 6

$$c_{min} \leq ks_{min} + \lceil \lg m \rceil + 1$$

Proof: By properties 9,10,11 all the pairs of rows of G^k are covered by the union of SC_i, for $i = 1, ..., k$ and C_{2m}, hence forming an identifying code. Since by property 9 every SC_i is a set cover of the original problem and $|C_{2m}| = 1 + \lceil \lg m \rceil$, the Lemma follows. ∎

Suppose next that there is a polynomial time algorithm that approximates the identifying code within a performance ratio $\sigma = (1 - \epsilon') \ln n$ for some $\epsilon' > 0$. We can apply it on G^k, with $k = \ln^2 m$, to get an approximation of size at most $c \leq \sigma c_{\text{min}} \leq \sigma k s_{\text{min}} \left(1 + \frac{\lceil \lg m \rceil}{k s_{\text{min}}}\right) \leq \sigma k s_{\text{min}} \left(1 + O(\ln^{-1} m)\right)$. By property 9 we can select then the minimum size SC_i as our approximation to the original set cover problem,

$$SC^* = \arg \min_{SC_{i|1 \leq i \leq k}} |SC_i|,$$

whose size is at most $sc = |SC^*| \leq \frac{c}{k}$. Hence the performance ratio of our set cover algorithm is:

$$\begin{aligned} \frac{sc}{s_{\text{min}}} &\leq (1 - \epsilon') \ln n \left(1 + O(\ln^{-1} m)\right) \\ &= (1 - \epsilon') \ln \left(m(2 \ln^2 m + 1)\right) \left(1 + O(\ln^{-1} m)\right) \\ &\leq (1 - \epsilon') \ln m \left(1 + O\left(\frac{\ln \ln m}{\ln m}\right)\right) \left(1 + O(\ln^{-1} m)\right), \end{aligned}$$

and for large enough m we can write for some $\epsilon > 0$

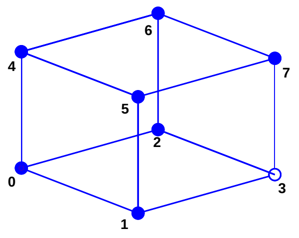
$$\frac{sc}{s_{\text{min}}} \leq (1 - \epsilon) \ln m$$

contradicting [28]. ∎

D. An identifying code approximation with tight guarantees

The identifying codes problem is actually a special case of the test cover problem [40]. Recall that a test cover problem asks to find the smallest set $\mathbf{T}$ of tests $T_i \subset \mathbf{U}$ such that any pair $x, y \in \mathbf{U}$ is distinguished by at least one test T_i (*i.e.*, $|\{x, y\} \cap T_i| = 1$). Then simply consider the base set to be the set of vertices of the graph, *i.e.*, $\mathbf{U} = V$, and its outgoing balls as the set of tests, $\mathbf{T} = \{B^-(v) | \forall v \in V\}$. Every pair of vertices will be distinguished by a code if and only if the corresponding set of tests constitutes a test cover of $(\mathbf{U}, \mathbf{T})$. It follows that test cover approximations can be applied to produce "good" identifying codes. One such greedy approximation was recently devised by Berman et al. [29] using a modified notion of entropy as the optimization measure. This greedy approximation was proven to have a performance ratio of $1 + \ln n$, where n is the number of elements in the base set. Applying this algorithm to graphs of size n guarantees identifying codes with the same performance ratio, closing the gap (up to a small constant) from the lower bound of Theorem 4.

Although this performance guarantee outperforms our set-cover based approximation, it is not obvious how to generalize the algorithm of Berman et al. to robust identifying codes. In the next section we discuss a natural way of generalizing our identifying code approximation of Theorem 1 to robust identifying codes.



$$\begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

Fig. 7. A 1-robust identifying code (codewords are solid circles) and the graph's adjacency matrix. The identifying set of vertex 1 is $\{0, 1, 5\}$.

VI. Robust Identifying Codes and The Set Multicover Problem

In this section, we provide a polynomial-time greedy approximation of the NP-complete [16, 20] robust identifying code problem. The graphs considered in this section are undirected graphs hence $B(v) = B^{+}(v) = B^{-}(v)$. Our approximation is based on an efficient reduction to the set k-multicover problem, for which a greedy approximation is known.

The *minimum set k-multicover* problem is a natural generalization of the set cover problem, in which given $(\mathbf{U}, \mathcal{S})$ we are seeking the smallest subset of $\mathcal{S}$ that covers every element in $\mathbf{U}$ at least k times (more formal definitions are in Section VI-B). Often this problem is addressed as a special case of the *covering integer problem* [36]. The set k-multicover problem admits a similar greedy heuristic to the set cover problem, with a corresponding performance guarantee [36] of at most $1 + \log\left(\max_{S_i \in \mathcal{S}}(|S_i|)\right)$.

Recall that a non-empty subset $\mathbb{C} \subseteq V$ is called a *code*, and for a given code $\mathbb{C}$, the *identifying set* $I_{\mathbb{C}}(v)$ of a vertex v is defined to be the codewords directed towards v, *i.e.*, $I_{\mathbb{C}}(v) = B(v) \cap \mathbb{C}$ (if $\mathbb{C}$ is not specified, it is assumed to be the set of all vertices V). A code $\mathbb{C}$ is an *identifying code* if each identifying set of the code is unique, in other words

$$\forall u, v \in V \quad u = v \longleftrightarrow I_{\mathbb{C}}(u) = I_{\mathbb{C}}(v).$$

Robust identifying codes were suggested in [4] for applications of location detection in harsh environments where nodes (codewords) are likely to fail. Intuitively an r-robust identifying code, $\mathbb{C}$, maintains its identification property even when any r codewords are either inserted or removed from $\mathbb{C}$.

Definition 2 *An identifying code $\mathbb{C}$ over a given graph $G = (V, E)$ is said to be r-robust if $I_{\mathbb{C}}(u) \oplus A \neq I_{\mathbb{C}}(v) \oplus D$ for all $v \neq u$ and $A, D \subset V$ with $|A|, |D| \leq r$. Here $\oplus$ denotes the symmetric difference.*

Consider a three dimensional cube as in Figure 7 and let $\mathbb{C} = \{0, 1, 2, 4, 5, 6, 7\}$. Clearly, the identifying sets are all unique, and hence the code is an identifying code. A closer look reveals that $\mathbb{C}$ is actually a 1-robust identifying code, so that it remains an identifying code even upon removal or insertion of any codeword into any identifying set.

Note that this definition differs from the *vertex* robust identifying codes of [6], where it is additionally required that $|I_{\mathbb{C}}(v)| \geq r$ for all $v \in V$.

A. Reduction intuition

Similarly to Section V we turn to the graph's adjacency matrix to provide an alternate perspective to the identifying code problem. Recall that in this matrix, rows or columns are *characteristic vectors* of corresponding balls, in that their i-th entries are 1 if the i-th element of U is in the corresponding ball. Selecting codewords is equivalent to selecting columns to form a *code* matrix of size $n \times |C|$. A code is thus identifying if the *Hamming distance* between every two rows in the code matrix is at least one (recall that the Hamming distance of two binary vectors is the number of ones in their bitwise XOR). Similarly, if the Hamming distance of every two rows in the code matrix is at least $2r + 1$ then the set of vertices is r-robust.

We next form the $\frac{n(n-1)}{2} \times n$ *difference* matrix by stacking the bitwise XOR results of every two different rows in the adjacency matrix. The problem of finding a minimum size r-robust identifying code is equivalent to selecting a minimum number of columns to form a code matrix for which the Hamming distance between any pair of distinct rows is at least $2r+1$. Or equivalently: selecting the minimum number of columns in the difference matrix such that all rows in the resulting matrix have Hamming weight of at least $2r+1$. This equivalent problem is nothing but a set $(2r+1)$-multicover problem, if one regards the columns of the difference matrix as the characteristic vectors of a family of subsets $\mathcal{S}$ over the base set of all pairs of rows in the adjacency matrix.

In the next subsection we formalize this intuition into a rigorous reduction.

B. Reduction

In this section, we formally reduce the problem of finding the smallest sized r-robust identifying code over an arbitrary graph G to a $(2r+1)$-multicover problem.

Formally we connect the following problems:

c) SET MULTI-COVER (SC_k):

`INSTANCE:`	Subsets S of U, an integer $k \geq 1$.
`SOLUTION:`	$S' \subseteq S$ such that for every element $u \in U$, $\lvert\{s \in S' : u \in s\}\rvert \geq k$.
`MEASURE:`	The size of the multicover: $\lvert S'\rvert$.

d) Robust ID-CODE (rID):

`INSTANCE:`	Graph $G = (V, E)$, and integer $r \geq 0$.
`SOLUTION:`	An r-robust identifying code $C \subseteq V$.
`MEASURE:`	The size $\lvert C\rvert$.

Theorem 5 *Given a graph G of n vertices, finding an r-robust identifying code requires no more computations than a $(2r+1)$-multicover solution over a base set of $\frac{n(n-1)}{2}$ elements together with $O(n^3)$ operations of length n binary vectors.*

Recall that the *difference set* $D_{\mathbb{C}}(u,v)$ is the symmetric difference between the identifying sets of vertices $u, v \in V$. In addition, recall that the *distinguishing set* δ_c is the set of vertex pairs in U for which c is a member of their difference set where $U = \{(u,z) | u \neq z \in V\}$.

As indicated in [4], a code is r-robust if and only if the size of the smallest difference set is at least $2r+1$. Equivalently, a code is r-robust if and only if its distinguishing sets $(2r+1)$-multicover all the pairs of vertices in the graph.

Lemma 7 *Given $G = (V, E)$ the following statements are equivalent:*

1) $\mathbb{C} = \{c_1, ..., c_k\}$ *is an r-robust identifying code.*
2) $|D_{\mathbb{C}}(u,v)| \geq 2r+1$*, for all* $u \neq v \in V$
3) *The set* $\{\delta_{c_1}, ..., \delta_{c_k}\}$ *forms a* $(2r+1)$*-multicover of* $U = \{(u,v) \,|\, \forall\ u \neq v \in V\}$.

Proof of Theorem 5: Let `SMC` be a set multicover algorithm and consider the following construction of an r-robust identifying code.

```
rID$(G, r) \to \mathbb{C}$
1. Compute $\{I(u) | u \in V\}$.
2. Compute $\Delta = \{\delta_u | u \in V\}$.
3. $\mathbf{C} \leftarrow$ SMC$(2r+1, U, \Delta)$
4. Output $\mathbb{C} \leftarrow \{u \in V | \delta_u \in \mathbf{C}\}$
```

The resulting code, $\mathbb{C}$, is guaranteed by Lemma 7 to be an r-robust identifying code, and the optimality of the set cover in step 3 guarantees that no smaller identifying code can be found. To complete the proof we observe that computing the identifying sets $I(u)$ naively requires $\Theta(n^2)$ additions of binary vectors, and computing Δ requires n operations for each of the $\frac{n(n-1)}{2}$ elements in U. ∎

Algorithm 2 Greedy robust ID-code $\texttt{rID} - \texttt{GREEDY}$

Let $G = (V, E)$ be a given graph, and let $\texttt{SMC} - \texttt{GREEDY}(k, \mathbf{U}, \mathcal{S})$ be the greedy set multicover algorithm, then given integer r the r-robust identifying code greedy algorithm is: $\texttt{rID} - \texttt{GREEDY}(G, r) \rightarrow \mathbb{C}_{greedy}$

```
1) Compute $\{I(u)|u \in V\}$.
2) Compute $\Delta = \{\delta_u|u \in V\}$.
3) $\mathbf{C} \leftarrow \texttt{SMC} - \texttt{GREEDY}(2r + 1, U, \Delta)$
4) Output $\mathbb{C}_{greedy} \leftarrow \{u \in V|\delta_u \in \mathbf{C}\}$
```

C. Approximation algorithm

The set multicover problem admits a greedy solution: in each iteration select the set which covers the maximum number of non k-multicovered elements. We use this heuristic together with Theorem 5 to introduce an r-robust identifying code algorithm (Algorithm 2).

It is well known [36] that the performance guarantee of the heuristic of the set multicover problem (defined as the ratio of the sizes of the greedy and minimum multicovers) is upper bounded by $1 + \ln \alpha$, where α is the largest set's size, namely

$$\frac{|\texttt{SMC} - \texttt{GREEDY}(k, \mathbf{U}, \mathcal{S})|}{|\texttt{SMC}(k, \mathbf{U}, \mathcal{S})|} \leq 1 + \ln \alpha \tag{9}$$

for any set multicover problem $(k, \mathbf{U}, \mathcal{S})$.

Theorem 6 *Given a graph $G = (V, E)$, of n nodes, let c_{greedy}, c_{min} be the sizes of the greedy ($\texttt{rID} - \texttt{GREEDY}$) and minimum r-robust identifying codes, respectively, then*

$$\frac{c_{greedy}}{c_{min}} \leq 1 + \log n + \log b_{max}$$

where b_{max} is the ball's size closest to $\frac{n}{2}$

Proof: δ_v contains all pairs of vertices, which v appear exactly in one of their incoming balls. Therefore v distinguishes between all pairs having one vertex in v's outgoing ball, $B(v)$, and the second from $V - B(v)$. Hence the size of a distinguishing set is given by $|\delta_v| = |B(v)|\,(n - |B(v)|)$. It is easy to see that the vertex with the largest distinguishing set is the one whose outgoing ball size is closest to $\frac{n}{2}$. Denote this outgoing ball by b_{max}, and based on Theorem 5 and $\texttt{rID} - \texttt{GREEDY}$ algorithm plug the size of its distinguishing set into the bound of Equation (9) to get

$$\begin{aligned} \frac{c_{\text{greedy}}}{c_{\text{min}}} &\leq 1 + \log(n - b_{max}) + \log b_{max} \\ &< 1 + \log n + \log b_{max}. \end{aligned}$$

∎

D. Localized robust identifying code and its approximation

It was observed in [4, 5] that an r-robust identifying code can be built in a localized manner, where each vertex only considers its two-hop neighborhood. This localization is possible when the identifying codes are required to produce only non-empty identifying sets[5]. In this section and henceforth we introduce this requirement, and call the resulting codes - *localized identifying codes*. These codes and their approximation algorithm are critical to the development of the distributed algorithms of the next section. Note that by definition the localized identifying codes are also dominating sets.

[5]In fact this definition of identifying codes is the traditional definition (*e.g.,* [1])

Algorithm 3 Localized r-robust code $\texttt{rID}-\texttt{LOCAL}(r,G)$

We start with a graph $G=(V,E)$ and a non-negative integer r. The greedy set multicover approximation is denoted $\texttt{SMC}-\texttt{GREEDY}(k,\mathbf{U},\mathcal{S})$.

```
1) Compute $\{D(u,v)|u \in V,\ v \in B(u;2)\}$
2) Compute $\Delta^2 = \{\delta_u^2 | u \in V\}$.
3) $\mathbf{C} \leftarrow \texttt{SMC}-\texttt{GREEDY}(2r+1,\ U^2, \Delta^2)$
4) Output $\mathbb{C}_{local} \leftarrow \{u \in V | \delta_u^2 \in \mathbf{C}\}$
```

Let $G=(V,E)$ be an undirected graph; we define the distance metric $\rho(u,v)$ to be the number of edges along the shortest path from vertex u to v. The ball of radius l around v is denoted $B(v;l)$ and defined to be $\{w \in V | \rho(w,v) \leq l\}$. So far we encountered balls of radius $l=1$, which we simply denoted by $B(v)$.

Recall that a vertex cover (or dominating set) is a set of vertices, such that the union of their balls of radius 1 covers V. We further extend this notion and define an *r-dominating* set as a set of vertices, which r-multicovers V (these sets are closely related to μ-fold 1-covering codes [42]).

Note that the definition of difference sets and hence identifying codes still involves only radius one balls, unlike the radius $(l>1)$ identifying codes discussed in [1].

Lemma 8 *Given a graph $G=(V,E)$, an $(r+1)$-dominating set $\mathbb{C}$ is also an r-robust identifying code if and only if $|D_{\mathbb{C}}(u,v)| \geq 2r+1$ for all $u,v \in V$ such that $\rho(u,v) \leq 2$.*

Proof: The forward implication is an application of Lemma 7. For the reverse implication we take $\mathbb{C}$ to be an $r+1$ dominating set and assume that $|D_{\mathbb{C}}(u,v)| \geq 2r+1$ for $\rho(u,v) \leq 2$; we will show that this assumption is also valid for $\rho(u,v) > 2$. This is because, for $\rho(u,v) > 2$, we have that $B(v) \cap B(u) = \emptyset$, meaning that $|D_{\mathbb{C}}(u,v)| = |B(v) \cap \mathbb{C}| + |B(u) \cap \mathbb{C}|$. Since $\mathbb{C}$ is an $r+1$ dominating set, it must be that $|B(y) \cap \mathbb{C}| \geq r+1$ for all vertices y, giving that $|D_{\mathbb{C}}(u,v)| > 2r+1$. Applying Lemma 7 we thus see that $\mathbb{C}$ must be r-robust. ∎

The localized robust identifying code approximation Lemma 8 can serve as the basis for a reduction from an identifying code problem to a set cover problem, similar to the one in Theorem 5. The main difference is that we will restrict base elements to vertex pairs that are at most two hops apart, and we then need to guarantee that the resulting code is still r-robust.

Towards this end we define $U^2 = \{(u,v) \mid \rho(u,v) \leq 2\}$, the set of all pairs of vertices (including (v,v)) that are at most two hops apart. Similarly, we will localize the distinguishing set δ_v to U^2 as follows:

$$\delta_v^2 = (\delta_v \cap U^2) \cup \{(u,u) | u \in B(v)\},$$

The resulting *localized identifying code approximation* is thus given by Algorithm 3 and can be shown to provide an r-robust identifying code for any graph that admits one (we omit the proof due to space considerations).

Theorem 7 *Given an undirected graph $G=(V,E)$ of n vertices, the performance ratio* $\texttt{rID}-\texttt{LOCAL}$ *is upper bounded by:*

$$\frac{c_{greedy}}{c_{min}} < \ln \gamma + 1,$$

where $\gamma = \max_{v \in V} |B(v)|(|B(v;3)| - |B(v)| + 1)$.

Proof: Recall that δ_v includes all vertex pairs where one vertex is in the outgoing ball of v and the other is not. The size of the intersection of δ_v and U^2 can be upper bounded by $|B(v)|(|B(v;3)| - |B(v)|)$, which becomes an equality when all vertices in $B(v;3)$ are within distance 2 of each other. Finally,

δ_v^2 includes also $|B(v)|$ pairs of the same vertex (u, u). Therefore $|\delta_v^2| \leq |B(v)|(|B(v;3)| - |B(v)| + 1)$, which at its maximum, can be applied to the performance guarantee of Equation (9) to complete the proof. ▮

In the next subsection we present a distributed implementation of the identifying code localized approximation. The following lemma supplements Lemma 8 by providing additional "localization". At the heart of this lemma lies the fact that each codeword distinguishes between its neighbors and the remaining vertices.

Lemma 9 *The distinguishing sets δ_v^2 and δ_u^2 are disjoint for every pair (u, v) with $\rho(u, v) > 4$.*

Proof: Clearly, δ_v^2 includes all vertex pairs $(x, y) \in U^2$ where x is a neighbor of v and y is not. More precisely, $(x, y) \in \delta_v^2$ if

$$x \in B(v) \text{ and } y \in B(x;2) - B(v). \tag{10}$$

Moreover, for all such (x, y), $\rho(x, v) \leq 1$ and $\rho(y, v) \leq 3$. On the other hand, for $(x', y') \in \delta_u^2$ with $\rho(u, v) > 4$, either x' or y' must be a neighbor of u, and hence of distance > 3 from v. Thus, δ_u^2 and δ_v^2 are disjoint. ▮

Lemma 9 implies that when applying the greedy algorithm, a decision on choosing a codeword only affects decisions on vertices within four hops; the algorithm is thus localized to vicinities of balls of radius four.

VII. Distributed Approximation to robust identifying codes

Several parallel algorithms exist in the literature for set cover and for the more general covering integer programs (*e.g.,* [35]). There are also numerous distributed algorithms for finding a minimum (connected) dominating set based on set cover and other well known approximations such as linear programming relaxation (*e.g.,* [43]). In a recent work, Kuhn et al. [44] devised a distributed algorithm for finding a dominating set with a constant runtime. The distributed algorithm uses a design parameter which provides a tradeoff between the runtime and performance.

Unfortunately, the fundamental assumption of these algorithms is that the elements of the base set are independent computational entities (*i.e.,* the nodes in the network); this makes it non-trivial to apply them in our case, where elements correspond to pairs of nodes that can be several hops apart. Moreover, we assume that the nodes are energy constrained so that reducing communications is very desirable, even at the expense of longer execution times and reduced performance.

We next provide two distributed algorithms. The first is completely asynchronous, guarantees a performance ratio of at most $\ln \gamma + 1$, and requires $\Theta(c_{\text{dist}})$ iterations at worst, where c_{dist} is the size of the identifying code returned by the distributed algorithm and $\gamma = \max_{v \in V} |B(v)|(|B(v;3)| - |B(v)| + 1|)$. The second is a randomized algorithm, which requires a coarse synchronization, guarantees a performance ratio of at most $\ln \gamma + 1$, and for some arbitrarily small $\epsilon > 0$ operates within $O\left(\frac{\gamma n^{\frac{K+2+\epsilon}{K-1}}}{K}\right)$ time slots (resulting in $O(c_{\text{dist}} \max_{v \in V} |B(v;4|))$ messages). The value $K \geq 2$ is a design parameter that trades between the size of the resulting r-robust identifying code and the required number of time slots to complete the procedure.

In the next subsection we describe the model and initialization stages that are common to both distributed algorithms.

A. Model and initialization

With a model similar to [4], we assume that every vertex (node) in a graph is an independent processing entity that is pre-assigned a unique serial number and can communicate reliably and collision-freely

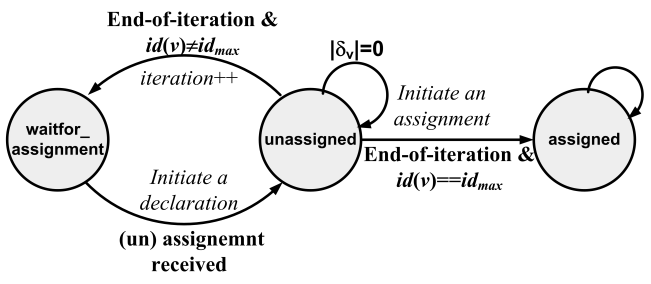


Fig. 8. Asynchronous distributed algorithm state diagram in node $v \in V$

(perhaps using higher-layer protocols) over a shared medium with its immediate neighborhood. In [4] the entities were sensors and the edges in the graph represented RF connectivity. Every node can determine its neighborhood from the IDs on received transmissions, and higher radius balls can be determined by multi-hop protocols. In our case, we will need to know $G(v;4)$ - the subgraph induced by all vertices of distance at most four from v.

Our distributed algorithms are based on the fact that, by definition, each node v can distinguish between the pairs of nodes which appear in its corresponding distinguishing set δ_v^2 that initially can be determined by (10). This distinguishing set is updated by removing pairs that get covered $2r+1$ times (*i.e.*, that have $2r+1$ elements in their difference set), as new codewords are added to the identifying code being constructed; their presence is advertised by flooding their four-hop neighborhood.

B. The asynchronous algorithm $\mathtt{rID-ASYNC}$

The state diagram of the asynchronous distributed algorithm is shown in Figure 8. All nodes are initially in the *unassigned* state, and transitions are effected according to messages received from a node's four-hop neighborhood. Two types of messages can accompany a transition: *assignment* and *declaration* messages, with the former indicating that the initiating node has transitioned to the *assigned* state, and the latter being used to transmit data. Both types of messages also include five fields: the *type*, which is either "assignment" or "declaration", the *ID* identifying the initiating node, the *hop* number, the *iteration* number, and *data*, which contains the size of the distinguishing set in the case of a declaration message.

Following the initialization stage, every node declares its distinguishing set's size. As a node's declaration message propagates through its four hop neighborhood, every forwarding node updates two internal variables, ID_{max} and δ_{max}, representing the ID and size of the *most distinguishing* node (ties are broken in favor of the lowest ID). Hence, when a node aggregates the declaration messages initiated by all its four hop neighbors (we say that the node reached its *end-of-iteration* event), ID_{max} should hold the most distinguishing node in its four hop neighborhood. A node that reaches an end-of-iteration event transitions to either the *waitfor_assignment* state or to the final *assigned* state depending if it is the most distinguishing node.

The operation of the algorithm is completely asynchronous; nodes take action according to their state and messages received. During the iterations stage, nodes initiate a declaration message only if they receive an assignment message or if an updated declaration (called an *unassignment* message) is received from the most distinguishing node of the previous iteration. All messages are forwarded (and their hop number is increased) if the hop number is less than four. To reduce communications load, a mechanism for detecting and eliminating looping messages should be applied.

Every node, v, terminates in either an "unassigned" state with $|\delta_v^2| = 0$ or in the "assigned" state. Clearly, nodes that terminate in the "assigned" state constitute a localized r-robust identifying code.

1) Performance evaluation:

Algorithm 4 Asynchronous r-robust algorithm ($\texttt{rID} - \texttt{ASYNC}$)

We start with a graph G, with vertices labeled by *ID*, and a non-negative integer r. The following distributed algorithm run at node $v \in V$ produces an r-robust identifying code.

```
PRECOMP:
  • Compute $\delta_v^2$ using (10).
  • Initiate a declaration message and set state = "unassigned".
  • Set $ID_{max} = ID(v)$, $\delta_{max} = |\delta_v^2|$, and $ms$ to be an empty assignment message.
ITERATION:
  • Increment $hop(ms)$ and forward all messages of $hop(ms) < 4$.
  • if received an assignment message $ms$ with $state \neq assigned$ then
      – Update $\delta_v^2$ by removing all pairs covered $2r+1$ times.
      – Initiate a declaration message and set state = "unassigned".
      – Reinitialize $ID_{max} = ID(v)$ and $\delta_{max} = |\delta_v^2|$.
  • if $state = waitfor\_assignment$ and received an *un-assignment* message then initiate a declaration message.
  • if received a declaration message $ms$ with $state \neq assigned$ then
      – if $\delta_{max} < data(ms)$ or ($\delta_{max} = data(ms)$ and $ID_{max} > ID(ms)$) then $\delta_{max} = data(ms)$, $ID_{max} = ID(ms)$
  • if *end-of-iteration* reached then,
      – if $ID_{max} = ID(v)$ and $|\delta_v^2| > 0$ then $state = assigned$, initiate an assignment message.
      – otherwise $state = waitfor\_assignment$.
```

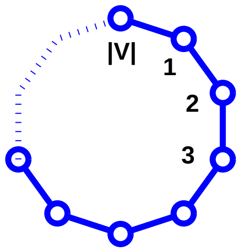


Fig. 9. Worst case topology for the asynchronous distributed r-robust algorithm in terms of runtime.

Theorem 8 *Algorithm 4 ($\texttt{rID} - \texttt{ASYNC}$) requires $\Theta(c_{dist})$ iterations and has a performance ratio*

$$\frac{c_{dist}}{c_{min}} < \ln \gamma + 1,$$

where $\gamma = \max_{v \in V} |B(v)|(|B(v;3)| - |B(v)| + 1|)$.

The first part of the Theorem follows from Theorem 7 and the fact that only the most distinguishing sets in a four hop neighborhoods is assigned to be a codeword. To see the number of iterations of the algorithm, we first note that in each iteration at least one codeword is assigned. The case of a cycle graph (Fig. 9) demonstrates that, in the worst case, exactly one node is assigned per iteration.

It follows that the amount of communications required in the iteration stage is $\Theta(c_{dist}|V|\max(|B(v;4)|))$, which can be a significant load for a battery powered sensor network. This can be significantly reduced if some level of synchronization among the nodes is allowed. In the next section we suggest a synchronized distributed algorithm that eliminates declaration messages altogether.

Algorithm 5 Synchronous r-robust algorithm ($\mathtt{rID}-\mathtt{SYNC}$)

We start with a graph G and non-negative integer r. The following distributed algorithm run at node $v \in V$ produces an r-robust identifying code.

```
PRECOMP:
  • Set: $slot = s_F$, $subslot = L$, $state = unassigned$.
  • Calculate the assignment slot.
ITERATE: while $state = unassigned$ and $slot \geq s_1$ do,
  • $l = random\{1, ..., L\}$
  • if received assignment message, $ms$ then,
      – if $hop(ms) < 4$ forward $ms$ with $hop(ms) + +$.
      – Recalculate the assignment slot.
  • else if $subslot = l$ and $slot = as$ then,
      – $state = assigned$
      – Transmit $ms$ with $id(ms) = id(v)$, and $hop(ms) = 1$
```

C. A low-communications randomized algorithm $\mathtt{rID}-\mathtt{SYNC}$

In this subsection, we assume that a coarse time synchronization among vertices within a neighborhood of radius four can be achieved. In particular, we will assume that the vertices maintain a basic time *slot*, which is divided into L *subslots*. Each subslot duration is longer than the time required for a four hop one-way communication together with synchronization uncertainty and local clock drift. After an initialization phase, the distributed algorithm operates on a time *frame*, which consists of F slots arranged in decreasing fashion from s_F to s_1. In general, F should be at least as large as the largest distinguishing set (*e.g.*, $F = \frac{n(n-1)}{2}$ will always work). A frame synchronization within a neighborhood of radius four completes the initialization stage.

The frame synchronization enables us to eliminate all the declaration messages of the asynchronous algorithm. Recall that the declaration messages were required to perform two tasks: (i) determine the most distinguishing node in its four hop neighborhood, and (ii) form an iteration boundary, *i.e.*, end-of-iteration event. The second task is naturally fulfilled by maintaining the slot synchronization. The first task is performed using the frame synchronization: every node maintains a synchronized slot counter, which corresponds to the size of the current *most distinguishing* node. If the slot counter reaches the size of a node's distinguishing set, the node assigns itself to the code. The subslots are used to randomly break ties.

1) Iterations stage: Each iteration takes place in one time slot, starting from slot s_F. During a slot period, a node may transmit a message ms indicating that it is assigning itself as a codeword; the message will have two fields: the identification number of the initiating node, $id(ms)$, and the hop number, $hop(ms)$. A node assigns itself to be a codeword if its *assignment time*, which refers to a slot as and subslot l, has been reached. Every time an assignment message is received, the assignment slot as of a node is updated to match the size of its distinguishing set; the assignment subslot is determined randomly and uniformly at the beginning of every slot.

2) Performance evaluation: Algorithm 5 ($\mathtt{rID}-\mathtt{SYNC}$) requires at most $O(n^2)$ slots ($O(Ln^2)$ subslots), though it can be reduced to $O(L\gamma)$ if the maximum size of a distinguishing set is propagated throughout the network in the precomputation phase. The communications load is low (*i.e.*, $O(c_{\text{dist}} \cdot \max_{v\in V}(|B(v;4)|))$), and includes only assignment messages, which are propagated to four hop neighborhoods.

In the case of ties, $\mathtt{rID}-\mathtt{SYNC}$ can provide a larger code than gained from the localized approximation. This is because ties in the distributed algorithm are broken arbitrarily, and there is a positive probability (shrinking as the number of subslots L increases) that more than one node will choose the same subslot within a four hop neighborhood. As such, L is a design parameter that provides a tradeoff between performance ratio guarantees and the runtime of the algorithm as suggested in the following theorem.

Theorem 9 *For asymptotically large graphs, Algorithm* $\mathtt{rID-SYNC}$ *guarantees with high probability a performance ratio of*

$$\frac{c_{dist}}{c_{min}} < K(\ln\gamma + 1),$$

where $\gamma = \max_{v\in V} |B(v)|(|B(v;3)| - |B(v)| + 1|)$*. The algorithm also requires* $O\left(\frac{\gamma n^{\frac{K+2+\epsilon}{K-1}}}{K}\right)$ *subslots to complete for design parameter* $K \geq 2$ *and arbitrarily small* $\epsilon > 0$*.*

Proof: If no more than K tied nodes assign themselves simultaneously on every assignment slot, then we can upper bound the performance ratio by a factor K of the bound in Theorem 7, as in the theorem statement. We next determine the number of subslots L needed to guarantee the above assumption asymptotically with high probability.

Let $P(K)$ denote the probability that no more than K tied nodes assign themselves in every assignment slot. Clearly, $P(K) \geq (1-\bar{p}(K))^{c_{\text{dist}}}$, where $\bar{p}(K)$ is the probability that, when t nodes are assigned independently and uniformly to L subslots, there are at least $K < t$ assignments to the same subslot. One can see that

$$\begin{aligned}\bar{p}(K) &= \sum_{k=K}^{t} L\binom{t}{k}L^{-k}\left(1-\frac{1}{L}\right)^{t-k} \\ &\leq \sum_{k=K}^{t}\binom{t}{k}L^{1-k} \leq \sum_{k=K}^{t} L\left(\frac{te}{Lk}\right)^{k} \\ &\leq tL\left(\frac{te}{LK}\right)^{K},\end{aligned}$$

for e being the natural logarithm and based on the assumption that $\frac{te}{LK} < 1$. Let $t = c_{\text{dist}} = n$ (this only loosens the bound) and $L = \frac{e}{K}n^{\frac{K+2+\epsilon}{K-1}}$. Then,

$$\begin{aligned}P(K) &\geq \left(1 - tL\left(\frac{te}{LK}\right)^{K}\right)^{c_{\text{dist}}} \\ &\geq \left(1 - \frac{e}{K}\frac{1}{n^{1+\epsilon}}\right)^{n} \to 1.\end{aligned}$$

∎

VIII. Simulations

Here we have simulated the centralized, localized and synchronized distributed identifying code algorithms, and applied them to grids (*e.g.,* [6]), random graphs with different edge probabilities [14], and to geometric random graphs with different nodes densities [31]. As a performance measure, we use the averaged size of the identifying code. For the case of $r = 0$ (*i.e.,* simple identifying code) the simulation results are compared to the algorithm suggested by Ray et al. in [4]. In addition, we show a combinatorial lower bound derived first by Karpovsky et al. in [1], and the asymptotic result (in n - the size of the graph) of Moncel et al. [14], who showed that an arbitrary set of a threshold number of codewords is an identifying code with high probability, and that this number is asymptotically tight.

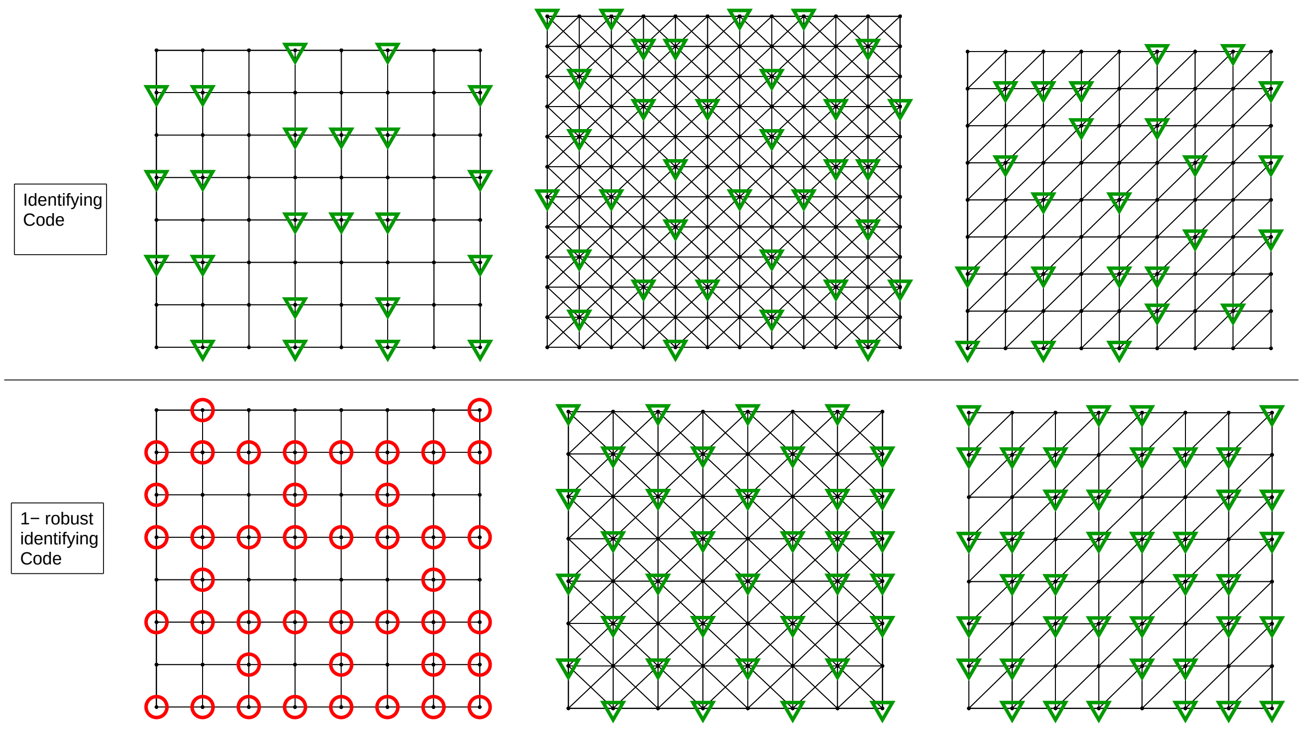


Fig. 10. Examples of efficient constructions of identifying tiles for the square, king and triangular infinite grids produced by the localized, `rID − LOCAL` (triangles), and centralized, `rID − GREEDY` (circles), algorithms. The constructions densities can be found in Table I.

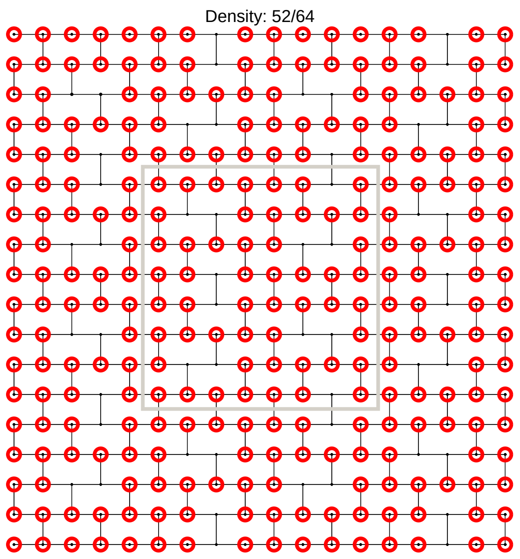
A. *Infinite grids*

Significant amount of the study efforts into identifying codes were focused on infinite grids [1, 6]. Many of these studies use tilings to suggest constructive upper bounds on the size of an identifying codes. Table I compares these tight constructions and bounds with the best outcomes of the centralized, localized and the distributed algorithms when applied to the square, king, and triangle infinite grids. Infinite grids were emulated by the use of finite girds with wrapped around edges (torii). The size of the torus limits the size of the largest tile of the infinite code, and therefore has a large impact on the resulting codes. The algorithms outcomes were also varied by randomly relabeling the torus nodes in between runs. Table I summarizes the best results achieved for the different torii sizes, and their matching constructions appear in Figure 10. Surprisingly, the localized (`rID − LOCAL`) algorithm outperformed the centralized (`rID − GREEDY`) for almost all grids and torii sizes, coming pretty close to the results in [6]. One reason for this behavior may be that the additional requirement that the code will also be dominating is not significant when it comes to grids. Therefore the centralized algorithm takes into account many irrelevant vertices pairs that may influence the optimization step.

Finally, Figure 11 shows that it is sometimes possible to get very good codes for small graphs through a pruned exhaustive search (in this case, an adaptation of the branch-and-bound algorithm [45]). The 1-robust identifying codes in the figure for the hexagonal and triangular finite grids actually have a density slightly better than those reported in [6].

B. *(Geometric) random graphs*

Fig. 12(a) shows the theoretic lower bound and the results of the centralized greedy algorithm. It can be seen that a significant enhancement in performance over the algorithm devised by Ray et al. is achieved. It should be noted that as n grows the curves for basically any algorithm should converge to Moncel's



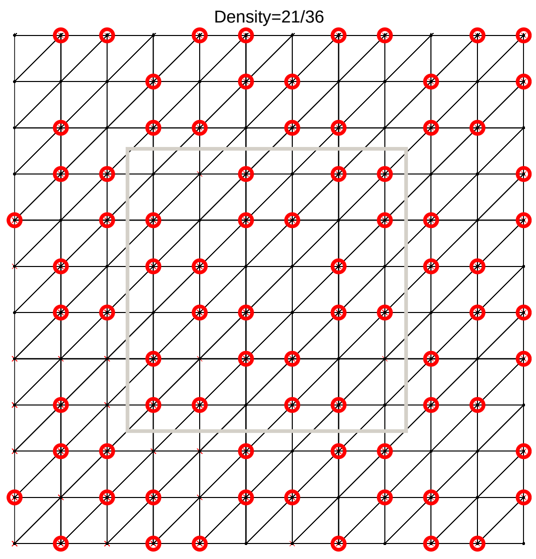


Fig. 11. Branch and bound algorithm results for 1-robust identifying codes for the hexagonal (left) and triangular (right) grids. The red circles represent codewords, and the tile is captured by a gray square.

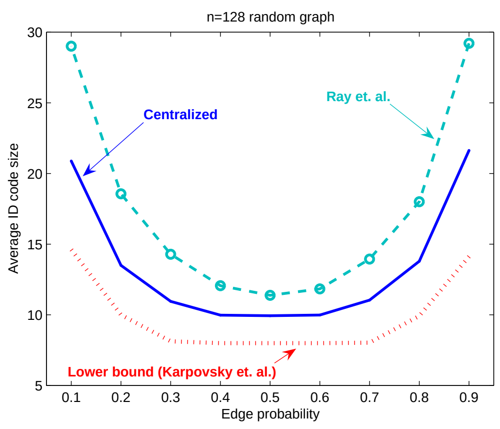
TABLE I
DENSITIES OF IDENTIFYING CODES FOR INFINITE GRIDS TAKEN FROM [6] VS. SIMULATION RESULTS USING OUR APPROXIMATIONS (IN BRACKETS).

	square	king	triangular
ID code	$\frac{112}{320}$ $(\frac{115}{320})$	$\frac{16}{72}$ $(\frac{17}{72})$	$\frac{27}{108}$ $(\frac{32}{108})$
1-Robust	$\leq \frac{40}{64}$ $(\frac{43}{64})$	$\frac{32}{64}$ $(\frac{35}{64})$	$\leq \frac{24}{40}$ $(\frac{25}{40})$

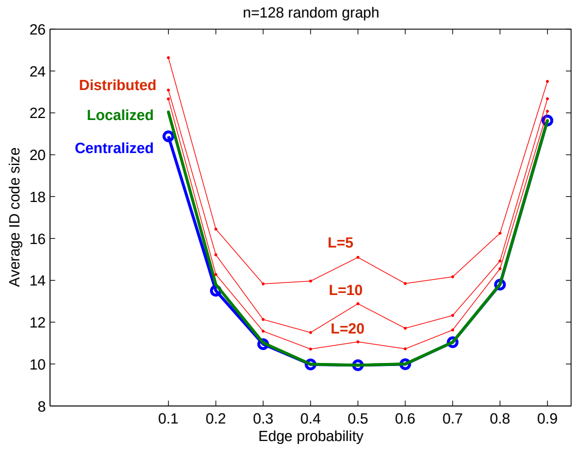
asymptotic result, as illustrated in Fig. 13(a). Still, the convergence rate appears to be very slow, suggesting that for reasonably large networks there is a lot to gain from the suggested algorithms compared to the simple approach of arbitrarily picking a code, whose size satisfies the threshold number of [14].

Fig. 12(b) shows the simulation results for the localized and distributed algorithms compared to the centralized one. Recall that the performance of the asynchronous algorithm, $\texttt{rID}-\texttt{ASYNC}$, is identical to the localized approximation. It can be observed that the results of the localized algorithm nearly match the results of the centralized algorithms. Divergence is evident for low edge probabilities, where it is harder to find a dominating set. Recall that there is a tradeoff between performance and the runtime of the synchronized distributed algorithm, $\texttt{rID}-\texttt{SYNC}$. The smaller the number of subslots parameter, L, the shorter the runtime and the larger the degradation in performance due to unresolved ties. Degradation in performance is also more evident when ties are more likely to happen, *i.e.*, when the edge probability is approaching 0.5. The results of the centralized r-robust identifying code algorithm are shown in Figure 12(c).

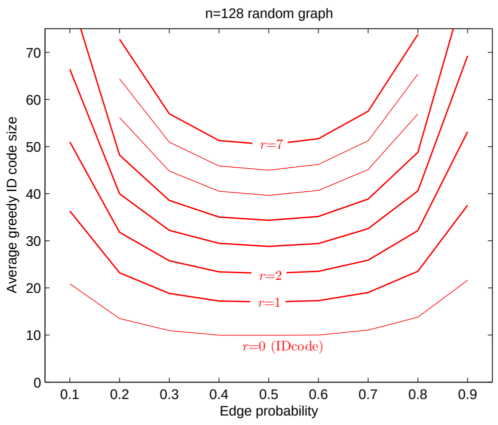
Fig. 13(b) shows the codeword density for geometric random graphs using the localized and distributed approaches, and the fraction of such graphs admitting an identifying code. It also presents the largest fraction of *indistinguishable* nodes, *i.e.*, nodes that share the same identifying set, obtained in the simulation. As can be seen, the localized and distributed approaches (with $L = 10$) yield very similar code sizes. The fraction of graphs admitting identifying codes is rather small (less than half the graphs) even for high node densities. However, the size of indistinguishable nodes sets is still small, suggesting that the system's reduction in *resolution*, i.e. the geometric size of the largest indistinguishable set, is not too



(a) Centralized greedy algorithm



(b) Localized (and rID − ASYNC) and distributed algorithm, rID − SYNC, for different subslot (L) values



(c) Centralized r-robust identifying codes algorithm.

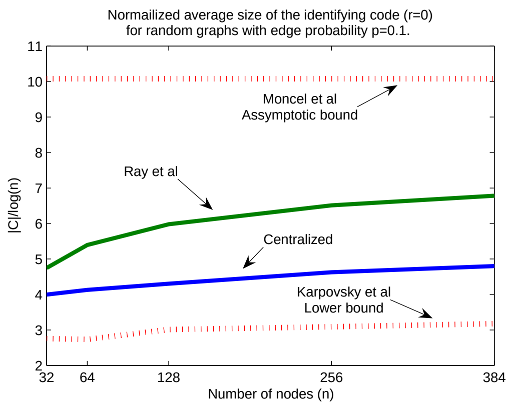
Fig. 12. Average size of the minimum identifying code for random graphs with edge probability p, and $n = 128$ vertices.

high. It should be noted that approaches such that of ID-CODE [4] are not designed to cope with graphs which do not have identifying codes, resulting in a code of all vertices.

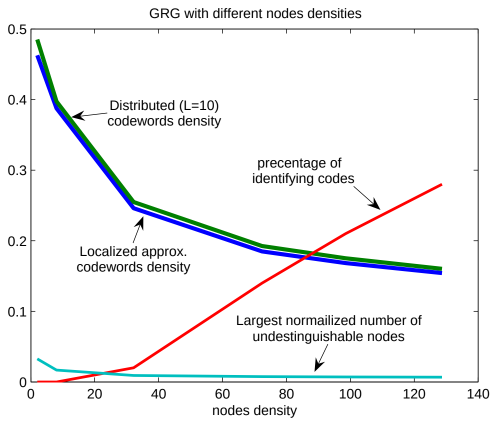
IX. Conclusions

We have provided connections between identifying codes problems and covering problems, for which greedy approximation algorithms are known. The connections to traditional identifying codes have been sufficiently size-preserving to produce an approximation algorithm, and should be useful for the number of identifying code applications that have been developed in the literature. In addition, we have provided general hardness of approximation results that show that our approximations are very close to the best possible. In particular, for the case of (non robust) identifying codes we show that a known approximation achieves up to a small constant multiple of the best possible performance achievable by any polynomial approximation (under commonly used complexity assumptions). It remains an open question if our approximation algorithms for the $(r \geq 1)$-robust identifying codes are similarly near-optimal.

We have also demonstrated by simulations that the proposed algorithms outperform the existing approximation algorithm of [4] and that they come close to efficient combinatorial constructions of robust identifying codes for two dimensional grids, which have been well-studied in the literature. We have further provided two flavors of distributed algorithms that may be used for practical implementations in network applications.



(a) Average size of the simple ($r = 0$) identifying code for random graphs with edge probability $p = 0.1$, and different number of vertices.



(b) Normalized size of the code for the localized (and $\texttt{rID}-\texttt{ASYNC}$) and distributed $\texttt{rID}-\texttt{SYNC}$ algorithm, fraction of graphs admitting an identifying code, and maximum fraction of indistinguishable nodes for geometric random graphs with different node densities.

Fig. 13. Additional simulation results for (geometric) random graphs.

ACKNOWLEDGMENT

The anonymous reviewers at the IEEE Transactions on Information Theory have provided thorough and significant improvements that have been incorporated into this paper.

REFERENCES

[1] M. G. Karpovsky, K. Chakrabarty, and L. B. Levitin, “A new class of codes for identification of vertices in graphs,” *IEEE Transactions on Information Theory*, vol. 44, no. 2, pp. 599–611, March 1998.

[2] S. Ray, R. Ungrangsi, F. D. Pellegrinin, A. Trachtenberg, and D. Starobinski, “Robust location detection in emergency sensor networks,” *Proceedings INFOCOM*, April 2003.

[3] R. Ungrangsi, A. Trachtenberg, and D. Starobinski, “An implementation of indoor location detection systems based on identifying codes,” *Proceedings of the IFIP International Conference on Intelligence in Communication Systems (INTELLCOMM 04)*, 2004.

[4] S. Ray, D. Starobinski, A. Trachtenberg, and R. Ungrangsi, “Robust location detection with sensor networks,” *IEEE Journal on Selected Areas in Communications (Special Issue on Fundamental Performance Limits of Wireless Sensor Networks)*, vol. 22, no. 6, August 2004.

[5] M. Laifenfeld and A. Trachtenberg, “Disjoint identifying codes for arbitrary graphs,” *IEEE International Symposium on Information Theory, Adelaide, Australia*, 4-9 Sept 2005.

[6] I. Honkala, M. Karpovsky, and L. Levitin, “On robust and dynamic identifying codes,” *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 599–612, February 2006.

[7] T. Y. Berger-Wolf, W. E. Hart, and J. Saia, “Discrete sensor placement problems in distribution networks,” *Journal of Mathematical and Computer Modelling*, vol. 42, no. 13, pp. 1385–1396, Dec 2005.

[8] M. Laifenfeld, A. Trachtenberg, R. Cohen, and D. Starobinski, “Joint monitoring and routing in wireless sensor networks using robust identifying codes,” *IEEE Broadnets 2007*, September 2007.

[9] M. Laifenfeld, A. Trachtenberg, and T. Berger-Wolf, “Identifying codes and the set cover problem,” *Proceedings of the 44th Annual Allerton Conference on Communication, Control, and Computing*, Sept 2006.

[10] S. Gravier and J. Moncel, “Construction of codes identifying sets of vertices,” *The Electronic Journal of Combinatorics*, vol. 12, no. 1, 2005.

[11] I. Honkala and A. Lobstein, “On identifying codes in binary Hamming spaces,” *Journal of Combinatorial Theory, Series A*, vol. 99, no. 2, pp. 232–243, 2002.

[12] Y. Ben-Haim and S. Litsyn, “Exact minimum density of codes identifying vertices in the square grid,” *SIAM Journal of Discrete Mathematics*, vol. 19, no. 1, pp. 69–82, 2005.

[13] S. R. T. Laihonen, “Codes identifying sets of vertices,” *Lecture Notes in Computer Science 2227*, 2001.

[14] J. Moncel, A. Frieze, R. Martin, M. Ruszink, and C. Smyth, “Identifying codes in random networks,” *IEEE International Symposium on Information Theory, Adelaide, 4-9 Sept.*, 2005.

[15] P. Slater, “Fault-tolerant locating-dominating sets,” *Discrete Mathematics*, vol. 249, pp. 179–189, 2002.

[16] I. Charon, O. Hudry, and A. Lobstein, “Identifying and locating-dominating codes: NP-completeness results for directed graphs,” *IEEE Transactions on Information Theory*, vol. 48, no. 8, pp. 2192–2200, August 2002.

[17] S. Gravier, R. Klasing, and J. Moncel, “Hardness results and approximation algorithms for identifying codes and locating-dominating codes in graphs,” LaBRI, Technical Report RR-1417-06, November 2006.

[18] J. Suomela, "Approximability of identifying codes and locating–dominating codes," *Information Processing Letters*, vol. 103, no. 1, pp. 28–33, 2007.

[19] K. B. Lakshmanan, D. J. Rosenkrantz, and S. S. Ravi, "Alarm placement in systems with fault propagation," *Theoretical Computer Science*, vol. 243, no. 1-2, pp. 269–288, 2000.

[20] I. Charon, O. Hudry, and A. Lobstein, "Minimizing the size of an identifying or locating-dominating code in a graph is NP-hard," *Theoretical Computer Science*, vol. 290, no. 3, pp. 2109–2120, 2003.

[21] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*. MIT Press, 2001.

[22] I. Charon, S. Gravier, O. Hudry, A. Lobstein, M. Mollard, and J. Moncel, "A linear algorithm for minimum 1-identifying codes in oriented trees," *Discrete Applied Mathematics*, vol. 154, no. 8, pp. 1246–1253, 2006.

[23] G. D. Cohen, I. S. Honkala, A. Lobstein, and G. Zémor, "New bounds for codes identifying vertices in graphs." *Electronic Journal of Combinatorics*, vol. 6, 1999.

[24] I. Charon, I. S. Honkala, O. Hudry, and A. Lobstein, "General bounds for identifying codes in some infinite regular graphs." *Electronic Journal of Combinatorics*, vol. 8, no. 1, 2001.

[25] I. Charon, O. Hudry, and A. Lobstein, "Identifying codes with small radius in some infinite regular graphs." *Electronic Journal of Combinatorics*, vol. 9, no. 1, 2002.

[26] I. Charon, I. S. Honkala, O. Hudry, and A. Lobstein, "The minimum density of an identifying code in the king lattice." *Discrete Mathematics*, vol. 276, no. 1-3, pp. 95–109, 2004.

[27] J. Moncel, "On graphs of n vertices having an identifying code of cardinality $\lceil log_2(n+1)\rceil$," *Discrete Applied Mathematics*, vol. 154, no. 14, pp. 2032–2039, 2006.

[28] U. Feige, "A threshold of ln n for approximating set cover," *Journal of the ACM*, vol. 45, no. 4, pp. 634–652, 1998.

[29] P. Berman, B. DasGupta, and M.-Y. Kao, "Tight approximability results for test set problems in bioinformatics," *Journal of Cmputer and System Sciences*, vol. 71, no. 2, pp. 145–162, 2005.

[30] R. Raz and S. Safra, "A sub-constant error-probability low-degree test, and a sub-constant error-probability pcp characterization of np," in *STOC '97: Proceedings of the twenty-ninth annual ACM Symposium on Theory of Computing*. New York, NY, USA: ACM Press, 1997, pp. 475–484.

[31] T. Müller and J.-S. Sereni, "Identifying and locating-dominating codes in (random) geometric networks," 2007, submitted to Combinatorics, Probability and Computing. ITI Series 2006-323 and KAM-DIMATIA Series 2006-797.

[32] L. Lovasz, "On the ratio of optimal integral and fractional covers," *Discrete Mathematics*, vol. 13, pp. 383–390, 1975.

[33] D. S. Johnson, "Approximation algorithms for combinatorial problems," *Journal of Computer and System Sciences*, vol. 9, pp. 256–278, 1974.

[34] P. Slavik, "A tight analysis of the greedy algorithm for set cover," *In Proceedings of the 28th Annual ACM Symposium on Theory of Computing (Philadelphia, Pa.)*, pp. 435–439, May 1996.

[35] S. Rajagopalan and V. Vazirani, "Primal-dual RNC approximation algorithms for set cover and covering integer programs," *SIAM Journal on Computing*, vol. 28, pp. 525–540, 1998.

[36] V. Vazirani, *Approximation Algorithms*. Springer-Verlag, July 2001.

[37] K. M. J. D. Bontridder, B. J. Lageweg, J. K. Lenstra, J. B. Orlin, and L. Stougie, "Branch-and-bound algorithms for the test cover problem," in *ESA '02: Proceedings of the 10th Annual European Symposium on Algorithms*. London, UK: Springer-Verlag, 2002, pp. 223–233.

[38] K. M. J. D. Bontridder, B. V. Halldrsson, M. M. Halldrsson, C. A. J. Hurkens, J. K. Lenstra, R. Ravi, and L. Stougie, "Approximation algorithms for the test cover problem," *Mathematical Programming-B 98*, no. 1-3, pp. 477–491, 2003.

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

[40] B. M. E. Moret and H. D. Shapiro, "On minimizing a set of tests," *SIAM Journal of Scientific & Statistical Computing.*, vol. 6, pp. 983–1003, 1985.

[41] M. Naor, L. J. Schulman, and A. Srinivasan, "Splitters and near-optimal derandomization," in *IEEE Symposium on Foundations of Computer Science*, 1995, pp. 182–191. [Online]. Available: citeseer.ist.psu.edu/naor95splitters.html

[42] G. Cohen, I. Honkala, S. Litsyn, and A. Lobstein, *Covering Codes*. Elsevier, 1997.

[43] Y. Bartal, J. W. Byers, and D. Raz, "Global optimization using local information with applications to flow control," in *IEEE Symposium on Foundations of Computer Science*, 1997, pp. 303–312. [Online]. Available: citeseer.ist.psu.edu/bartal97global.html

[44] F. Kuhn and R. Wattenhofer, "Constant-time distributed dominating set approximation," *Proceedings of the 22nd ACM Symposium on Principles of Distributed Computing (PODC'03)*, July 2003. [Online]. Available: citeseer.ist.psu.edu/kuhn03constanttime.html

[45] M. Laifenfeld, "Coding for network applications: Robust identification and distributed resource allocation," Ph.D. dissertation, Boston University, 2007.