Closure solvability for network coding and secret sharing

Maximilien Gadouleau *Member, IEEE*

***Abstract*—Network coding is a new technique to transmit data through a network by letting the intermediate nodes combine the packets they receive. Given a network, the network coding solvability problem decides whether all the packets requested by the destinations can be transmitted. In this paper, we introduce a new approach to this problem. We define a closure operator on a digraph closely related to the network coding instance and we show that the constraints for network coding can all be expressed according to that closure operator. Thus, a solution for the network coding problem is equivalent to a so-called solution of the closure operator. We can then define the closure solvability problem in general, which surprisingly reduces to finding secret-sharing matroids when the closure operator is a matroid. Based on this reformulation, we can easily prove that any multiple unicast where each node receives at least as many arcs as there are sources is solvable by linear functions. We also give an alternative proof that any nontrivial multiple unicast with two source-receiver pairs is always solvable over all sufficiently large alphabets. Based on singular properties of the closure operator, we are able to generalise the way in which networks can be split into two distinct parts; we also provide a new way of identifying and removing useless nodes in a network. We also introduce the concept of network sharing, where one solvable network can be used to accommodate another solvable network coding instance. Finally, the guessing graph approach to network coding solvability is generalised to any closure operator, which yields bounds on the amount of information that can be transmitted through a network.**

***Index Terms*—Network coding, guessing games, closure operators, secret sharing, matroids.**

I. Introduction

Network coding [1] is a protocol which outperforms routing for multicast networks by letting the intermediate nodes manipulate the packets they receive. In particular, linear network coding [2] is optimal in the case of one source; however, it is not the case for multiple sources and destinations [3], [4]. Although for large dynamic networks, good heuristics such as random linear network coding [5], [6] can be used, maximizing the amount of information that can be transmitted over a static network is fundamental but very difficult in practice. Solving this problem by brute force, i.e. considering all possible operations at all nodes, is computationally prohibitive. Different alternative approaches have been proposed to tackle this problem, notably using matroids, information inequalities, and group theory [7], [8], [9], [10], [11], [12]. In this paper, we provide a new approach to tackle this problem based on a closure operator defined on a related digraph. Closure operators are fundamental and ubiquitous mathematical objects.

The guessing number of digraphs is a concept introduced in [13], which connects graph theory, network coding, and circuit complexity theory. In [13] it was proved that an instance of network coding with r sources and r sinks on an acyclic network (referred to as a multiple unicast network) is solvable over a given alphabet if and only if the guessing number of a related digraph is equal to r. Moreover, it is proved in [14], [15] that any network coding instance can be reduced into a multiple unicast network. Therefore, the guessing number is a direct criterion on the solvability of network coding. One of the main advantages of the guessing number approach is to remove the hierarchy between sources, intermediate nodes, and destinations. In [15], the guessing number is also used to disprove a long-standing open conjecture on circuit complexity. In [16], the guessing number of digraphs was studied, and bounds on the guessing number of some particular digraphs were derived. The guessing number is also equal to the so-called graph entropy [13], [17]. This allows us to use information inequalities [18] to derive upper bounds on the guessing number. The guessing number of undirected graphs is studied in [19]. Moreover, in [20], the guessing number is viewed as the independence number of an undirected graph related to the digraph and the alphabet.

Shamir introduced the so-called threshold secret sharing scheme in [21]. Suppose a sender wants to communicate a secret $a \in A$ to n parties, but that an eavesdropper may intercept $r-1$ of the transmitted messages. We then require that given any set of $r-1$ messages, the eavesdropper cannot obtain any information about the secret. On the other hand, any set of r messages allows to reconstruct the original secret a. The elegant technique consists of sending evaluations of a polynomial $p(x) = \sum_{i=0}^{r-1} p_i x^i$, with $p_0 = a$ and all the other coefficients chosen secretly at random, at n nonzero elements of A; this is evidently reminiscent of Reed-Solomon codes. The threshold scheme was then generalised to *ideal secret sharing schemes* with different access structures, i.e. different sets of trusted parties. Brickell and Davenport have proved that the access structure must be the family of spanning sets of a matroid; also any linearly representable matroid is a valid access structure [22]. However, there exist matroids (such as the Vámos matroid [23]) which are not valid access structures. For a given access structure (or equivalently, matroid), finding the scheme is equivalent to a representation by partitions [24].

In this paper, we introduce a closure operator on digraphs, and define the closure solvability problem for any closure

The author is with the School of Engineering and Computing Sciences, Durham University, Durham, UK. Email: m.r.gadouleau@durham.ac.uk.

This paper was presented in part at the LMS-EPSRC Durham Symposium "Graph Theory and Interactions," Durham, UK, July 2013.

operator. This yields the following contributions.

- First of all, this framework encompasses network coding and ideal secret sharing. In particular, network coding solvability is equivalent to the solvability of the closure operator of a digraph associated to the network. This framework then allows us to think of network coding solvability on a higher, more abstract level. The problem, which used to be about coding functions, is now a simplified problem about partitions.
- This approach is particularly elegant, in different aspects. Firstly, the adjacency relations of the graph, and hence the topology of the network, are not visible in the closure operator. Therefore, the closure operator filters out some unnecessary information from the graph. Secondly, it is striking that all along the paper, most proofs will be elementary, including those of far-reaching results. Thirdly, this framework highlights the relationship with matroids unveiled in [9], [25].
- Like the guessing number approach, the closure operator approach also gets rid of the source-intermediate node-destination hierarchy. The guessing graph machinery of [20] can then be easily generalised to any closure operator. In other words, the interesting aspects of the guessing number approach can all be recast and generalised in our framework.
- This approach then yields interesting results. First, it was shown in [20] that the entropy of a digraph is equal to the sum of the entropies of its strongly connected components. Thus, one can split the solvability problem of a digraph into multiple ones, one for each strongly connected component [20]. In this paper, we extend this way of splitting the problem by considering the closure operators induced by the subgraphs. We can easily exhibit a strongly connected digraph whose closure operator is disconnected, i.e. which can still be split into two smaller parts. More specifically, if the graph is strongly connected but its closure operator is disconnected, then we can exhibit a set of vertices which are simply useless and can be disregarded for solvability. Second, we can prove that any digraph whose closure operator has rank two is solvable. This means that any multiple unicast with two source-receiver pairs is solvable, unless there exists an easily spotted bottleneck in the network. This has already been proved in [26]; our proof is much shorter and highlights the relation with coding theory and designs. Third, we can prove that any network with minimum in-degree equal to the number of source-receiver pairs is solvable by linear functions over all sufficiently large alphabets of size equal to a large prime power. Fourth, we prove an equivalence between network coding solvability and index coding solvability. Finally, we show how the bidirectional union of digraphs can be viewed as network sharing.

The rest of the paper is organised as follows. In Section II, we review some useful background. In Section III, we define the closure solvability problem and prove that network coding solvability is equivalent to the solvability of a closure operator. We then use this conversion in Section IV to prove the solvability of different classes of networks. We investigate how to combine closure operators in Section V. We finally define the solvability graph in VI and study its properties.

II. Preliminaries

A. Closure operators

Throughout this paper, V is a set of n elements. A closure operator on V is a mapping $\mathrm{cl} : 2^V \to 2^V$ which satisfies the following properties [27, Chapter IV]. For any $X, Y \subseteq V$,

1) $X \subseteq \mathrm{cl}(X)$ (extensive);
2) if $X \subseteq Y$, then $\mathrm{cl}(X) \subseteq \mathrm{cl}(Y)$ (isotone);
3) $\mathrm{cl}(\mathrm{cl}(X)) = \mathrm{cl}(X)$ (idempotent).

A *closed set* is a set equal to its closure. For instance, in a group one may define the closure of a set as the subgroup generated by the elements of the set; the family of closed sets is simply the family of all subgroups of the group. Another example is given by linear spaces, where the closure of a set of vectors is the subspace they span.

A closure operator satisfies the following properties. For any $X, Y \subseteq V$,

1) $\mathrm{cl}(X)$ is equal to the intersection of all closed sets containing X;
2) $\mathrm{cl}(\mathrm{cl}(X) \cap \mathrm{cl}(Y)) = \mathrm{cl}(X) \cap \mathrm{cl}(Y)$, i.e. the family of closed sets is closed under intersection;
3) $\mathrm{cl}(X \cup Y) = \mathrm{cl}(\mathrm{cl}(X) \cup \mathrm{cl}(Y))$.
4) $X \subseteq \mathrm{cl}(Y)$ if and only if $\mathrm{cl}(X) \subseteq \mathrm{cl}(Y)$.

We refer to

$$r := \min\{|b| : \mathrm{cl}(b) = V\}$$

as the *rank* of the closure operator. For instance, in a linear space, this is the dimension of the space. Any set $b \subseteq V$ of size r and whose closure is V is referred to as a *basis* of cl.

An important class of closure operators are *matroids* [28], which satisfy the Mac Lane-Steinitz exchange property[1]: if $X \subseteq V$, $v \in V$ and $u \in \mathrm{cl}(X \cup v) \backslash \mathrm{cl}(X)$, then $v \in \mathrm{cl}(X \cup u)$. A special class consists of the uniform matroids, typically denoted as $U_{r,n}$, where

$$U_{r,n}(X) = \begin{cases} V & \text{if } |X| \geq r \\ X & \text{otherwise.} \end{cases}$$

Clearly, $U_{r,n}$ has rank r.

B. Functions and their kernels

While network coding typically works with functions assigned to vertices, it is elegant to work with *partitions* (for a review of their properties, the reader is invited to [29]). Recall that a partition of a set B is a collection of subsets, called parts, which are pairwise disjoint and whose union is the whole of B.For instance, the partition of B into $|B|$ singletons is he so-called *equality partition*, and is denoted as E_B. We denote the parts of a partition f as $P_i(f)$ for all i; thus, $P_i(E_B) = \{i\}$ for all $i \in B$.

[1]In order to simplify notation, we shall identify a singleton $\{v\}$ with its element v

If any part of f is contained in a unique part of g, we say f refines g. The equality partition refines any other partition, while the universal partition (the partition with one part) is refined by any other partition. The *common refinement* of two partitions f, g of B is given by $h := f \vee g$ with parts

$$P_{i,j}(h) = P_i(f) \cap P_j(g) : P_i(f) \cap P_j(g) \neq \emptyset.$$

We shall usually consider a tuple of n partitions $f = (f_1, \ldots, f_n)$ assigned to elements of a finite set V with n elements. In that case, for any $X \subseteq V$, we denote the common refinement of all $f_v, v \in X$ as $f_X := \bigvee_{v \in X} f_v$. For any $S, T \subseteq V$ we then have $f_{S \cup T} = f_S \vee f_T$.

Any function $\bar{f} : B \to C$ has a *kernel* denoted as $f := \{\bar{f}^{-1}(c) : c \in \bar{f}(B)\}$, defined by the partition of B into pre-images under $\bar{f}$. Conversely, any partition of B in at most $|C|$ can be viewed as the kernel of some function from B to C. Note that two functions $\bar{f}$, $\bar{g}$ have the same kernel if and only if $\bar{f} = \pi \circ \bar{g}$ for some permutation π of C.

C. Digraphs

Throughout this paper, we shall only consider digraphs [30] with no repeated arcs. Unless specified otherwise, the vertex set of the digraph will be V, a set of cardinality n. We shall denote the arc set as $E(D)$, since the letter A will be reserved for the alphabet. However, we do allow edges in both directions between two vertices, referred to as *bidirectional edges* (we shall abuse notations and identify a bidirectional edge with a corresponding undirected edge) and loops over vertices. In other words, the digraphs considered here are of the form $D = (V, E)$, where $E \subseteq V^2$. For any vertex v of D, its in-neighborhood is $v^- = \{u \in V : (u, v) \in E(D)\}$ and its in-degree is the size of its in-neighborhood. By extension, we denote $X^- = \bigcup_{v \in X} v^-$ for any set of vertices X. Also, by analogy, the out-neighbourhood of v is $v^+ := \{u \in V : (v, u) \in E(D)\}$. We say that a digraph is *strongly connected* if there is a path from any vertex to any other vertex of the digraph.

The *girth* of a digraph is the minimum length of a cycle, where we consider a bidirectional edge as a cycle of length 2. A digraph is *acyclic* if it has no directed cycles. In this case, we can order the vertices $v_1, \ldots, v_n$ so that $(v_i, v_j) \in E(D)$ only if $i < j$ (this is referred to as an acyclic ordering in [30]). The cardinality of a maximum induced acyclic subgraph of the digraph D is denoted as $\mathrm{mias}(D)$. A set of vertices X is a *feedback vertex set* if and only if any directed cycle of D intersects X, or equivalently if $V \backslash X$ induces an acyclic subgraph.

Definition 1: [20] For any digraphs D_1 and D_2 with disjoint vertex sets V_1 and V_2, we denote the disjoint union, unidirectional union, and bidirectional union of D_1 and D_2 as the graphs on $V_1 \cup V_2$ and respective edge sets

$$\begin{aligned} E(D_1 \cup D_2) &= E(D_1) \cup E(D_2) \\ E(D_1 \vec{\cup} D_2) &= E(D_1 \cup D_2) \cup \{(v_1, v_2) : v_1 \in V_1, v_2 \in V_2\} \\ E(D_1 \bar{\cup} D_2) &= E(D_1 \vec{\cup} D_2) \cup \{(v_2, v_1) : v_1 \in V_1, v_2 \in V_2\}. \end{aligned}$$

In other words, the disjoint union simply places the two graphs next to each other; the unidirectional union adds all possible arcs from D_1 to D_2 only; the bidirectional union adds all possible arcs between D_1 and D_2.

D. Guessing game and guessing number

A *configuration* on a digraph D on V over a finite alphabet A is simply an n-tuple $x = (x_1, \ldots, x_n) \in A^n$. A *protocol* $f = (f_1, \ldots, f_n)$ of D is a mapping $f : A^n \to A^n$ such that $f(x)$ is locally defined, i.e. $f_v(x) = f_v(x_{v^-})$ for all v. The fixed configurations of f are all the configurations $x \in A^n$ such that $f(x) = x$: $\mathrm{Fix}(f) = \{x \in A^n : f(x) = x\}$. The *guessing number* of D is then defined as the logarithm of the maximum number of configurations fixed by a protocol of D:

$$g(D, A) = \max_f \left\{ \log_{|A|} |\mathrm{Fix}(f)| \right\}.$$

We now review how to convert a multiple unicast problem in network coding to a guessing game. Note that any network coding instance can be converted into a multiple unicast without any loss of generality [14], [15]. We suppose that each sink requests an element from an alphabet A from a corresponding source. This network coding instance is *solvable* over A if all the demands of the sinks can be satisfied at the same time. We assume the network instance is given in its *circuit representation*, where each vertex represents a distinct coding function and hence the same message flows every edge coming out of the same vertex [15]; again this loses no generality. This circuit representation has r source nodes, r sink nodes, and m intermediate nodes. By merging each source with its corresponding sink node into one vertex, we form the digraph D on $n = r+m$ vertices. In general, we have $g(D, A) \leq r$ for all A and the original network coding instance is solvable over A if and only if $g(D, A) = r$ [15]. Note that the protocol on the digraph is equivalent to the coding and decoding functions on the original network.

For any digraphs D_1, D_2 on disjoint vertex sets V_1 and V_2 respectively, we have

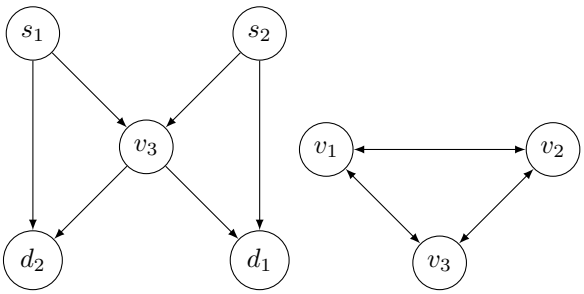
$$\begin{aligned} g(D_1 \cup D_2, A) &= g(D_1 \vec{\cup} D_2, A) = g(D_1, A) + g(D_2, A), \\ g(D_1 \bar{\cup} D_2, A) &\leq \min\{|V_1| + g(D_2, A), |V_2| + g(D_1, A)\}, \end{aligned}$$

for all alphabets A [20]. Notably, we can always consider strongly connected graphs only.

We illustrate the conversion of a network coding instance to a guessing game for the famous butterfly network in Figure 1. It is well-known that the butterfly network is solvable over all alphabets, and conversely it was shown that the clique K_3 has guessing number 2 over any alphabet. The combinations and decoding operations on the network are equivalent to the protocol on the digraph. For instance, if v_3 transmits the opposite of the sum of the two incoming messages modulo $|A|$ on the network, the corresponding protocol lets all nodes guess minus the sum modulo $|A|$ of their incoming elements.

E. Parameters of undirected graphs

Most results of this section are not directly interesting for network coding but will be instrumental in some of the proofs of the paper. As such, a first reading of the paper can safely skip this part.



(a) Network coding instance (b) Guessing game

Fig. 1. The butterfly network as a guessing game.

An independent set in a (simple, undirected) graph is a set of vertices where any two vertices are non-adjacent. The *independence number* $\alpha(G)$ of an undirected graph G is the maximum cardinality of an independent set. The *chromatic number* $\chi(G)$ of G is the smallest number of parts of a partition of its vertex set into independent sets [31]. An automorphism for a graph $G = (V, E)$ is a permutation ϕ of V such that $\{u, v\} \in E$ if and only if $\{\phi(u), \phi(v)\} \in E$. A graph G is *vertex-transitive* if for all $u, v \in V$, there is an automorphism ϕ of G such that $\phi(u) = v$. For a connected vertex-transitive graph which is neither an odd cycle nor a complete graph, we have [31, Corollary 7.5.2], [32].

$$\frac{|V(G)|}{\alpha(G)} \leq \chi(G) \leq (1 + \log \alpha(G)) \frac{|V(G)|}{\alpha(G)}. \tag{1}$$

We now review three types of products of graphs; all products of two graphs G_1 and G_2 have $V(G_1) \times V(G_2)$ as vertex set. We denote two adjacent vertices u and v in a graph as $u \sim v$.

1) In the *co-normal product* $G_1 \oplus G_2$, we have $(u_1, u_2) \sim (v_1, v_2)$ if and only if $u_1 \sim v_1$ or $u_2 \sim v_2$. We have $\alpha(G_1 \oplus G_2) = \alpha(G_1)\alpha(G_2)$.
2) In the *lexicographic product* (also called composition) $G_1 \cdot G_2$, we have $(u_1, u_2) \sim (v_1, v_2)$ if and only if either $u_1 = v_1$ and $u_2 \sim v_2$, or $u_1 \sim v_1$. Although this product is not commutative, we have $\alpha(G_1 \cdot G_2) = \alpha(G_1)\alpha(G_2)$.
3) In the *cartesian product* $G_1 \square G_2$, we have $(u_1, u_2) \sim (v_1, v_2)$ if and only if either $u_1 = v_1$ and $u_2 \sim v_2$, or $u_2 = v_2$ and $u_1 \sim v_1$. We have $\chi(G_1 \square G_2) = \max\{\chi(G_1), \chi(G_2)\}$ and $\alpha(G_1 \square G_2) \leq \min\{\alpha(G_1)|V(G_2)|, \alpha(G_2)|V(G_1)|\}$.

III. Closure solvability and network coding

A. Closure operators related to digraphs

Let D be a digraph on V, a set of n vertices.

Definition 2: The *D-closure* of a set of vertices X is defined as follows. We let $c_D(X) = X \cup \{v \in V : v^- \subseteq X\}$ and the D-closure of X is obtained by applying it c_D repeatedly n times: $\mathrm{cl}_D(X) := c_D^n(X)$.

This definition can be intuitively explained as follows. Suppose we assign a function to each vertex of D, which only depends on its in-neighbourhood (the function which decides which message the vertex will transmit). If we know the messages sent by the vertices of X, we also know the messages which will be sent by any vertex in $c_D(X)$. By applying this iteratively, we can determine all messages sent by the vertices in $\mathrm{cl}_D(X)$. Therefore, $\mathrm{cl}_D(X)$ represents everything that is determined by X.

We give an alternate, easier to manipulate, definition of the D-closure below.

Lemma 1: For any $X \subseteq V$, $Y = \mathrm{cl}_D(X) \backslash X$ is the largest set of vertices inducing an acyclic subgraph such that $Y^- \subseteq Y \cup X$.

Proof: First, it is clear that Y is a set of vertices inducing an acyclic subgraph such that $Y^- \subseteq Y \cup X$. Conversely, suppose Z induces an acyclic subgraph and $Z^- \subseteq Z \cup X$. Denoting $Z_0 = \emptyset$ and $Z_i = \{v \in Z : v^- \subseteq X \cup Z_{i-1}\}$ for $1 \leq i \leq n$, we have $Z_i \subseteq c_D^i(X) \backslash X$ and hence $Z = Z_n \subseteq Y$. ∎

Example 1: Some special classes of digraphs yield famous closure operators (all claims follow from Lemma 1).

1) If D is an acyclic digraph, then $\mathrm{cl}_D = U_{0,n}$, i.e. $\mathrm{cl}_D(X) = V$ for all X. This can be intuitively explained by the fact that an acyclic digraph comes from a network coding instance without any source or destination: no information can then be transmitted.
2) If D is the directed cycle C_n, then $\mathrm{cl}_{C_n} = U_{1,n}$, i.e. $\mathrm{cl}_D(\emptyset) = \emptyset$ and $\mathrm{cl}_D(v) = V$ for all $v \in V$. Therefore, the solutions are $(n, 1, n)$ MDS codes, such as the repetition code. Intuitively, C_n comes from a network coding instance with one source and one destination, and a chain of $n-1$ intermediate nodes each transmitting a message to the next until we reach the destination.
3) If D is the clique K_n, then $\mathrm{cl}_{K_n} = U_{n-1,n}$, i.e $\mathrm{cl}_D(X) = X$ if $|X| \leq n-2$ and $v \in \mathrm{cl}_D(V \backslash v)$ for all $v \in V$. Therefore, the solutions of cl_{K_n} are exactly $(n, n-1, 2)$ MDS codes, such as the parity-check code. Intuitively, K_n comes from a generalisation of the butterfly network, with one intermediate node receiving from all sources and transmitting to all destinations.
4) If D has a loop on each vertex, then $\mathrm{cl}_D = U_{n,n}$, i.e. $\mathrm{cl}_D(X) = X$ for all $X \subseteq V$. This comes from a network with a link from every source to its corresponding destination.

Since $\mathrm{cl}_D(X) = V$ if and only if X is a feedback vertex set of D, we obtain that cl_D has rank $r_D = n - \mathrm{mias}(D)$.

B. Closure solvability

We now define the closure solvability problem. The instance consists of a closure operator cl on V with rank r, and of a finite alphabet A with $|A| \geq 2$.

Definition 3: A *coding function* for (cl, A) is a family f of n partitions of A^r into at most $|A|$ parts such that $f_X = f_{\mathrm{cl}(X)}$ for all $X \subseteq V$.

The problem is to determine whether there exists a coding function for (cl, A) such that f_V has A^r parts. That is, we want to determine whether there exists an n-tuple $f = (f_1, \ldots, f_n)$ of partitions of A^r in at most $|A|$ parts such that

$$\begin{aligned} f_X &= f_{\mathrm{cl}(X)} \quad \text{for all } X \subseteq V, \\ f_V &= E_{A^r}. \end{aligned}$$

For any partition g of A^r, we define its entropy as

$$H(g) := r - |A|^{-r} \sum_i |P_i(g)| \log_{|A|} |P_i(g)|.$$

The equality partition on A^r is the only partition with full entropy r. Denoting $H_f(X) := H(f_X)$, we can recast the conditions above as

$$\begin{aligned} H_f(v) &\leq 1 \quad \text{for all } v \in V, \\ H_f(X) &= H_f(\mathrm{cl}(X)) \quad \text{for all } X \subseteq V, \\ H_f(V) &= r. \end{aligned}$$

Therefore, cl is solvable if and only if $H_f(V) = r$ for some coding function f of cl over A.

The first important case is solvability of uniform matroids, which is equivalent to the existence of MDS codes.

Proposition 1: For all r, n, and A, $U_{r,n}$ is solvable over A if and only if there exists an $(n, r, n-r+1)$-MDS code over an alphabet of cardinality $|A|$.

The proof follows the classical argument that a code of length n with cardinality $|A|^r$and minimum distance $n-r+1$ is separable (hence the term MDS code). We shall formally prove a much more general result in Section VI, therefore we omit the proof of Proposition 1.

In particular, a solution for $U_{2,n}$ is then equivalent to $n-2$ mutually orthogonal latin squares; they exist for all sufficient large alphabets. This illustrates the complexity of this problem: solving $U_{2,4}$ (i.e., determining the possible orders for two mutually orthogonal latin squares) was wrongly conjectured by Euler and solved in 1960 [33].

Combinatorial representations [25] were recently introduced in order to capture some of the dependency relations amongst functions. A solution for the uniform matroid corresponds to a combinatorial representation of its family of bases; however, in general this is not true. Indeed, any family of bases has a combinatorial representation, while we shall exhibit closure operators which are not solvable.

C. Closure solvability and network coding solvability

We consider a multiple unicast instance: an acyclic network N with r sources $s_1, \ldots, s_r$, r destinations $d_1, \ldots, d_r$, and m intermediate nodes, where each destination d_i requests the message x_i sent by s_i. We assume that the messages x_i, along with everything carried on one link, is an element of an alphabet A. Also, any vertex transmits the same message on all its outgoing links, i.e. we are using the circuit representation reviewed in Section II. We denote the cumulative coding functions at the nodes as $f = (f_1, \ldots, f_n)$, where the first r indeces correspond to the destinations and the other m indeces to the intermediate nodes, and $n = r + m$.

We now convert the network coding solvability problem into a closure solvability problem. Recall the digraph D on n vertices corresponding to the guessing game, reviewed in Section II.

Intuitively, if the destination d_i is able to recover x_i from the messages it receives, it is also able to recover any function $\sigma(x_i)$ of that message. Conversely, if it can recover $\pi(x_i)$ for some permutation π of A, then it can recover $x_i = \pi^{-1}(\pi(x_i))$ as well. We can then relax the condition and let d_i request any such $\pi(x_i)$. Viewing x_i as a function from A^r to A, sending $(x_1, \ldots, x_r)$ to x_i, we remark that $\pi(x_i)$ has the same kernel as x_i for any permutation π. Therefore, the correct relaxation is for d_i to request that the partition assigned to it be the same as that of the source s_i.

The relaxation above is one argument to consider partitions instead of functions. The second main argument is that the dependency relations are completely (and elegantly) expressed in terms of partitions, as illustrated in the proof of Theorem 1.

Theorem 1: The network N is solvable over A if and only if cl_D has rank r and is solvable over A.

Proof: Let $\bar{f}$ be a solution for N. Then it is easy to check that $f := (\ker \bar{f}_1, \ldots, \ker \bar{f}_n)$ is a family of partitions of A^r into at most $|A|$ parts such that $f_V = E_{A^r}$ and $f_{v \cup v^-} = f_{v^-}$. As such, $f_X = f_{c_D(X)}$ for all $X \subseteq V$ and hence $f_X = f_{\mathrm{cl}_D(X)}$.

Conversely, let f be a solution for cl_D over A and let $\bar{f}_v$ on N be any collection of functions with kernels $\ker \bar{f}_{s_i} = \ker \bar{f}_{d_i} = f_i$ for all $1 \leq i \leq r$ and $\ker \bar{f}_v = f_v$ for all $r+1 \leq v \leq n$. Since $f_{v \cup v^-} = f_{v^-}$, we have that $\bar{f}_v$ only depends on f_{v^-}; the number of parts of f_v indicates that $\bar{f}_v : A^n \to A$; finally, $f_{s_1,\ldots,s_r} = f_V = E_{A^r}$ indicates that $\bar{f}$ is a solution for N. ∎

We remark that the closure operator approach differs from Riis's guessing game approach. Although it also gets rid of the source/intermediate node/receiver hierarchy and works on the same digraph, the distinction is in the fact that now f corresponds to the cumulated coding functions.

IV. Main results

A. How to use closure solvability

So far, we have considered any possible closure operator. Let us reduce the scope of our study by generalising some concepts arising from matroid theory.

First, we say that a vertex is a *loop* if it belongs to the closure of the empty set. It is clear that removing $\mathrm{cl}(\emptyset)$ from V does not affect solvability (any vertex from $\mathrm{cl}(\emptyset)$ is useless). We therefore assume that $\mathrm{cl}(\emptyset) = \emptyset$. In particular, we only consider digraphs with positive minimum in-degree or in other words, that have a cycle.

Second, we say that cl is *separable* if for all $a, b \in V$ such that $a \notin \mathrm{cl}(b)$ and $b \notin \mathrm{cl}(a)$, we have $\mathrm{cl}(a) \cap \mathrm{cl}(b) = \emptyset$. Any matroid is separable; likewise it is easily seen that any D-closure is separable too. If cl is separable, we can further simplify the problem in a more general fashion than the so-called parallel elements in a matroid. There exists V' such that V is partitioned into parts $\{\mathrm{cl}(v') : v' \in V'\}$; for any $u \in V$, there exists $v' \in V'$ such that $\mathrm{cl}(u) \subseteq \mathrm{cl}(v')$. Again, considering the closure operator cl' on V' defined by $\mathrm{cl}'(X') = \mathrm{cl}(X') \cap V'$ does not affect solvability (since if cl is solvable, then there is a solution for cl where $f_u = f_{v'}$ for all $u \in \mathrm{cl}(v')$). Therefore, we can always restrict ourselves to D-closures where $\mathrm{cl}_D(v) = v$ for all v. In other words, we have just removed all vertices of in-degree one and by-passed them instead. Clearly, these vertices of degree one are useless for

network coding, as they do not bring any more combinations. The only thing they can do is forward the symbol they receive. As such, we might as well by-pass them.

There is a natural partial order on the family of closure operators of V. We denote $\mathrm{cl}_1 \leq \mathrm{cl}_2$ if for all X, $\mathrm{cl}_1(X) \subseteq \mathrm{cl}_2(X)$. This partial order has maximum element $U_{0,n}$ (with $\mathrm{cl}(X) = V$ for all $X \subseteq V$) and minimum element $U_{n,n}$ (where $\mathrm{cl}(X) = X$ for all X).

Any tuple f of partitions of A^r into at most $|A|$ parts naturally yields a closure operator on V: we define

$$\begin{aligned}\mathrm{cl}_f(X) &:= \{v \in V : f_{X\cup v} = f_X\} \\ &= \{v \in V : H_f(X \cup v) = H_f(X)\}.\end{aligned}$$

Proposition 2: f is a coding function for cl if and only if $\mathrm{cl} \leq \mathrm{cl}_f$. Therefore, if $\mathrm{cl}_1 \leq \mathrm{cl}_2$ have the same rank and cl_2 is solvable over A, then cl_1 is solvable over A.

Proof: If f is a coding function for cl, then $f_{\mathrm{cl}(X)} = f_{X\cup v} = f_X$ for all $v \in \mathrm{cl}(X)$ and hence $\mathrm{cl} \leq \mathrm{cl}_f$. Conversely, if $\mathrm{cl}(X) \subseteq \mathrm{cl}_f(X)$, then denote $\mathrm{cl}(X) = \{v_1, \ldots, v_k\}$ and $f_{\mathrm{cl}(X)} = f_{X\cup v_1} \vee f_{v_2,\ldots,v_k} = f_X \vee f_{v_2,\ldots,v_k} = \ldots = f_X$.

Since cl_2 is solvable, there exists a coding function f for cl_2 with entropy r, where r is the rank of cl_1 and cl_2. But then $\mathrm{cl}_1 \leq \mathrm{cl}_2 \leq \mathrm{cl}_f$ and hence f is also a solution for cl_1. ■

If cl is a matroid, the solvability problem is equivalent to determining whether they form a secret-sharing matroid, i.e. whether there exists a scheme whose access structure is the family of spanning sets of that matroid.

Theorem 2: If cl is a matroid, then cl is solvable over some alphabet if and only if it is a secret-sharing matroid.

Proof: By definition, a secret-sharing matroid is solvable over some alphabet. Conversely, let f be a solution for cl. Let rk be the rank function associated to cl, i.e. $\mathrm{rk}(X) = \min\{|b| : \mathrm{cl}(b) = \mathrm{cl}(X)\}$ and $\mathrm{cl}(X) = \{v \in V : \mathrm{rk}(X) = \mathrm{rk}(X \cup v)\}$ [28]. Then for any X, we have $H_f(X) = H_f(b) \leq |b| = \mathrm{rk}(X)$. Moreover, there exists Y such that $\mathrm{cl}(X \cup Y) = V$ and $\mathrm{rk}(Y) + \mathrm{rk}(X) = r$, hence $H_f(X) \geq H_f(V) - H_f(Y) \geq r - \mathrm{rk}(Y) \geq \mathrm{rk}(X)$. Thus, $H_f(X) = \mathrm{rk}(X)$ for all X and $\mathrm{cl}_f(X) = \{v \in V : \mathrm{rk}(X) = \mathrm{rk}(X \cup v)\} = \mathrm{cl}(X)$. ■

B. Solvable networks

In this subsection, we apply the conversion of network coding solvability in order to closure solvability to determine that some classes of networks are solvable. Using general closure operators allows us to think outside of networks. In particular, it allows us to use uniform matroids, which have been proved to be solvable over many alphabets (see Proposition 1), but which do not arise from networks in general (see Proposition 3 below).

Proposition 3: The uniform matroid $U_{r,n}$ is the D-closure of a digraph D if and only if $r \in \{0, 1, n-1, n\}$.

Proof: The cases $r = 0, 1, n-1, n$ respectively have been illustrated in Example 1. Conversely, suppose a digraph has D-closure $U_{r,n}$, where $2 \leq r \leq n-2$. Then any set of $n-r$ vertices induces an acyclic subgraph, while any set of $n-r+1$ vertices induces a cycle. This implies that any set of $n-r$ vertices induces a (directed) path. Without loss, let $v_1, \ldots, v_{n-r}$ induce a path (in that order), then $v_1, \ldots, v_{n-r}, v_{n-r+1}$ induce a cycle, and so do $v_1, \ldots, v_{n-r}, v_{n-r+2}$. Therefore, in the subgraph induced by $v_2, \ldots, v_{n-r+2}$, the vertex v_{n-r} has out-degree 2 and hence that graph is not a cycle. ■

We can then prove that all digraphs with minimum degree equal to the rank, or with rank 2, are solvable. Note that the case of rank 2 has already been proved in [26] using a much longer argument.

Theorem 3: Any cl_D of rank 2 is solvable over all sufficiently large alphabets. Moreover, if the mininum in-degree of D is equal to its rank, then cl_D is solvable by linear functions over all sufficiently large prime powers.

Proof: The simplifications above mean that we can assume $\mathrm{cl}_D(v) = v$ for all $v \in V$. This is equivalent to $\mathrm{cl}_D \leq U_{2,n}$, therefore by Proposition 2, any digraph with rank 2 is solvable whenever $U_{2,n}$ is.

Moreover, suppose the minimum in-degree is equal to the rank r. Then for any $X \subseteq V$ with $|X| \leq r$, we have $c_D(X) = X$ and hence $\mathrm{cl}_D(X) = X$; therefore, $\mathrm{cl}_D \leq U_{r,n}$. Again Proposition 2 yields the result. ■

V. Combining closure operators

In this section, we let $V_1, V_2 \subseteq V$ with respective cardinalities n_1 and n_2 such that $V_1 \cap V_2 = \emptyset$ and $V_1 \cup V_2 = V$. For any $X \subseteq V$, we denote $X_1 = X \cap V_1$ and $X_2 = X \cap V_2$. We also let $\mathrm{cl}_1, \mathrm{cl}_2$ be closure operators of rank r_1 and r_2 over V_1 and V_2, respectively.

A. Disjoint and unidirectional unions

We first generalise some definitions from matroid theory [28].

Definition 4: For any closure operator cl and any $V_2 \subseteq V$, the *deletion of* V_2 and the *contraction of* V_2 from cl are the closure operators defined on V_1 by

$$\begin{aligned}\mathrm{cl}\backslash^{V_2}(X) &:= \mathrm{cl}(X)\backslash V_2 \\ \mathrm{cl}/_{V_2}(X) &:= \mathrm{cl}(X \cup V_2)\backslash V_2\end{aligned}$$

for any $X \subseteq V_1$.

Proposition 4: If cl_D is the closure operator associated to the digraph D, then for any $V_2 \subseteq D$, $\mathrm{cl}_{D[V_1]} = \mathrm{cl}_D/_{V_2}$, where $D[V_1]$ is the digraph induced by the vertices in V_1. Thus $r(\mathrm{cl}_D/_{V_2}) = |V_1| - \mathrm{mias}(D[V_1])$ for any V_1.

Proof: Let $X \subseteq V_1$, then any subset Y of $V_1\backslash X = V\backslash(X \cup V_2)$ induces an acyclic subgraph of D if and only if it induces an acyclic subgraph of $D[V_1]$; moreover, $Y^- \subseteq X \cup Y$ in $D[V_1]$ if and only if $Y^- \subseteq X \cup Y \cup V_2$. By Lemma 1, we obtain $\mathrm{cl}_{D[V_1]}(X)\backslash X = \mathrm{cl}_D(X \cup V_2)\backslash(X \cup V_2)$ and hence $\mathrm{cl}_D/V_2(X) = \mathrm{cl}_{D[V_1]}(X)$. ■

Definition 5: The *disjoint union* and *unidirectional union* of cl_1 and cl_2 are closure operators on V respectively given by

$$\mathrm{cl}_1 \cup \mathrm{cl}_2(X) := \mathrm{cl}_1(X_1) \cup \mathrm{cl}_2(X_2)$$

$$\mathrm{cl}_1 \vec{\cup} \mathrm{cl}_2(X) := \begin{cases} V_1 \cup \mathrm{cl}_2(X_2) & \text{if } \mathrm{cl}_1(X_1) = V_1 \\ \mathrm{cl}_1(X_1) \cup X_2 & \text{otherwise.}\end{cases}$$

For any cl_1, cl_2 we have $\mathrm{cl}_1 \vec{\cup} \mathrm{cl}_2 \leq \mathrm{cl}_1 \cup \mathrm{cl}_2$ and

$$r(\mathrm{cl}_1 \cup \mathrm{cl}_2) = r(\mathrm{cl}_1 \vec{\cup} \mathrm{cl}_2) = r_1 + r_2.$$

Recall the definitions of unions of digraphs in Section II. Our definitions were tailored such that

$$\mathrm{cl}_{D_1 \cup D_2} = \mathrm{cl}_{D_1} \cup \mathrm{cl}_{D_2}$$
$$\mathrm{cl}_{D_1 \,\vec{\cup}\, D_2} = \mathrm{cl}_{D_1} \,\vec{\cup}\, \mathrm{cl}_{D_2}.$$

Moreover, if there is a loop on vertex v in the digraph D, then $\mathrm{cl}_D(X) = \mathrm{cl}_D(X \backslash v) \cup (X \cap v)$, or in other words, $\mathrm{cl}_D = \mathrm{cl}_{D[V\backslash v]} \cup U_{1,1} = \mathrm{cl}_{D[V\backslash v]} \,\vec{\cup}\, U_{1,1}$. We also remark that if cl_1 and cl_2 are matroids, then $\mathrm{cl}_1 \cup \mathrm{cl}_2$ is commonly referred to as the *direct sum* of cl_1 and cl_2 [28].

The disjoint and unidirectional unions are related to the contraction as follows.

Proposition 5: For any cl and any $V_2 \subseteq V$, the following are equivalent

1) $\mathrm{cl}/_{V_2} = \mathrm{cl}\backslash^{V_2}$, i.e. for all $X \subseteq V$, $\mathrm{cl}(X) \cap V_1 = \mathrm{cl}(X \cup V_2) \cap V_1$;
2) $\mathrm{cl}/_{V_2} \,\vec{\cup}\, \mathrm{cl}/_{V_1} \leq \mathrm{cl} \leq \mathrm{cl}/_{V_2} \cup \mathrm{cl}/_{V_1}$;
3) there exist $\mathrm{cl}_1, \mathrm{cl}_2$ defined on V_1 and V_2 respectively such that
$$\mathrm{cl}_1 \,\vec{\cup}\, \mathrm{cl}_2 \leq \mathrm{cl} \leq \mathrm{cl}_1 \cup \mathrm{cl}_2.$$

Proof: The first property implies the second, due to the following pair of inequalities: For any V_1,

$$\mathrm{cl}\backslash^{V_2} \,\vec{\cup}\, \mathrm{cl}/_{V_1} \leq \mathrm{cl} \leq \mathrm{cl}/_{V_2} \cup \mathrm{cl}/_{V_1}.$$

To prove the first inequality, we have

$$\mathrm{cl}\backslash^{V_2} \,\vec{\cup}\, \mathrm{cl}/_{V_1}(X) = V_1 \cup \mathrm{cl}/_{V_1}(X_2) = \mathrm{cl}(X_2 \cup V_1) \subseteq \mathrm{cl}(X)$$

if $V_1 \subseteq \mathrm{cl}(X_1)$ and

$$\mathrm{cl}\backslash^{V_2} \,\vec{\cup}\, \mathrm{cl}/_{V_1}(X) = (\mathrm{cl}(X_1) \cap V_1) \cup X_2 \subseteq \mathrm{cl}(X)$$

otherwise. For the second inequality, we have

$$\mathrm{cl}(X)\backslash V_2 = \mathrm{cl}(X_1 \cup X_2)\backslash V_2 \subseteq \mathrm{cl}(X_1 \cup V_2)\backslash V_2 = \mathrm{cl}/_{V_2}(X_1),$$

and similarly $\mathrm{cl}(X)\backslash V_1 \subseteq \mathrm{cl}/_{V_1}(X_2)$, and hence $\mathrm{cl}(X) \subseteq \mathrm{cl}/_{V_2} \cup \mathrm{cl}/_{V_1}(X)$.

Clearly, the second property implies the third one. Finally, if there exist such cl_1 and cl_2, then it is easy to check that $\mathrm{cl}_1 = \mathrm{cl}\backslash^{V_2} = \mathrm{cl}/_{V_2}$. ∎

B. Application to removing useless vertices

The first property of Proposition 5 indicates that V_2 has no effect on V_1; thus suggesting the following notation.

Definition 6: If there exists V_2 such that $\mathrm{cl}/_{V_2} = \mathrm{cl}\backslash^{V_2}$, we say that cl is disconnected and that V_2 is *weak*. If V_2 is weak and acyclic, then we say V_2 is *useless*.

The D-closure of a non strongly connected graph is disconnected. However, there are strongly connected graphs whose D-closure is disconnected.

Example 2 (Strongly connected graphs with disconnected D-closures): The canonical example of a strongly connected graph with disconnected closure operator is given in Figure 2 (a). On that graph, $V_2 = \{3\}$ is useless, for

$$\mathrm{cl}_D(\emptyset)\backslash 3 = \mathrm{cl}_D(3)\backslash 3 = \emptyset$$
$$\mathrm{cl}_D(1)\backslash 3 = \mathrm{cl}_D(13)\backslash 3 = 12$$
$$\mathrm{cl}_D(2)\backslash 3 = \mathrm{cl}_D(23)\backslash 3 = 12$$
$$\mathrm{cl}_D(12)\backslash 3 = \mathrm{cl}_D(123)\backslash 3 = 12$$

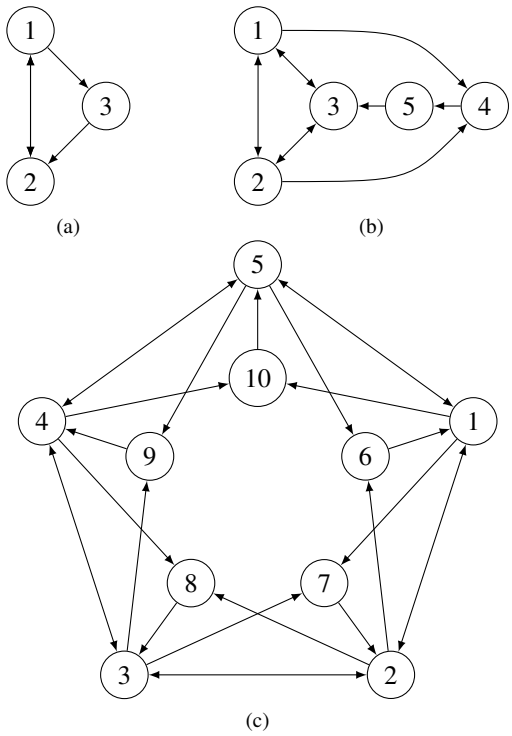


Fig. 2. Three strongly connected graphs with disconnected D-closures.

A larger example is given by the graph in Figure 2 (b), where $\mathrm{cl}_D\backslash^{V_2} = \mathrm{cl}_D/_{V_2}$ for $V_2 = \{4.5\}$. Each example leaves a graph which is solvable (a clique). On the other hand, in Figure 2 (c), the useless set is $V_2 = \{6, \ldots, 10\}$. Therefore, after removing the vertices 6 to 10, we are left with the undirected cycle of length 5, $\bar{C}_5$, which is not solvable [15].

Proposition 6: Suppose D is strongly connected, then V_2 is weak for cl_D if and only if it is useless.

Proof: We first claim that all arcs from V_2 to V_1 come from $\mathrm{cl}_{D[V_2]}(\emptyset)$. Indeed, let $u \in V_1$ such that $u^- \cap V_1 \neq \emptyset$. Then $u \in \mathrm{cl}_D/_{V_2}(V_1\backslash u) = \mathrm{cl}_D\backslash^{V_2}(V_1\backslash u)$, and hence $u \in \mathrm{cl}_D(V_1\backslash u)$. Since $u^- \subseteq \mathrm{cl}_D(V_1\backslash u)$, the intersection $X := \mathrm{cl}_D(V_1\backslash u) \cap V_2$ is not empty. By Lemma 1, X induces an acyclic subgraph and $X^- \subseteq V_1 \cap X$, which is equivalent to $X \subseteq \mathrm{cl}_{D[V_2]}(\emptyset)$.

Now, suppose V_2 is not acyclic, i.e. $V_2 \neq \mathrm{cl}_{D[V_2]}(\emptyset)$. But then, by the claim above there are no arcs from $V_2\backslash \mathrm{cl}_{D[V_2]}(\emptyset)$ to its complement, and D is not strongly connected. ∎

As a corollary, if D is an undirected graph, then cl_D is connected if and only if D is connected.

We remark that if V_2 is weak, then V_2 is closed, for $\mathrm{cl}(V_2)\backslash V_2 = \mathrm{cl}(\emptyset)\backslash V_2 = \emptyset$. Also, it is easy to check that the union of two weak sets is weak, hence there exists a largest weak set. Thus, if D is strongly connected, there exists a largest useless set, referred to as the *useless part* of D.

We say a cycle $v_1, \ldots, v_k$ is *chordless* if there does not exist $i, j \in \{1, \ldots, k\}$, $(i, j) \neq (1, k)$, such that $v_i, \ldots, v_j$ is a cycle. In other words, a chordless cycle does not cover another shorter cycle. Then $T(D)$ is the set of vertices which do not belong to any chordless cycle.

Theorem 4: Algorithm 1 removes the useless part of a strongly connected digraph in polynomial time.

Algorithm 1 Remove the useless part of a strongly connected digraph D

$T \leftarrow T(D)$
repeat
  $Found \leftarrow 0$
  while $v \in T$ **and** $Found = 0$ **do**
    $Found \leftarrow |v^-|$
    while $u \in v^-$ **and** $Found = 1$ **do** {Check that $\{v\}$ is useless}
      if $v \notin \mathrm{cl}_D(u^-\backslash v)$ **then**
        $Found = 0$
      end if
    end while
    if $Found > 0$ **then** {Remove v}
      $V \leftarrow V\backslash v$
      $T \leftarrow T\backslash v$
    end if
  end while
until $Found = 0$
return D

The proof of Theorem 4 is given in Appendix A.

C. Bidirectional union

Definition 7: The *bidirectional union* of cl_1 and cl_2 is defined as

$$\mathrm{cl}_1 \bar{\cup} \mathrm{cl}_2(X) := \begin{cases} V_1 \cup \mathrm{cl}_2(X_2) & \text{if } X_1 = V_1 \\ \mathrm{cl}_1(X_1) \cup V_2 & \text{if } X_2 = V_2 \\ X_1 \cup X_2 & \text{otherwise.} \end{cases}$$

It is easily shown that $\mathrm{cl}_1 \bar{\cup} \mathrm{cl}_2 \leq \mathrm{cl}_1 \vec{\cup} \mathrm{cl}_2$ and $r(\mathrm{cl}_1 \bar{\cup} \mathrm{cl}_2) = \min\{r_1 + n_2, r_2 + n_1\}$. Moreover, for any cl and any $V_1 \subseteq V$, we have

$$\mathrm{cl}/_{V_2} \bar{\cup} \mathrm{cl}/_{V_1} \leq \mathrm{cl} \leq \mathrm{cl}/_{V_2} \cup \mathrm{cl}/_{V_1}.$$

The first inequality means that the bidirectional union is the way to combine cl_1 and cl_2 which brings the fewest dependencies; as such it is the union of cl_1 and cl_2 with the highest entropy.

The bidirectional union of digraphs does correspond to the bidirectional union of closure operators:

$$\mathrm{cl}_{D_1 \bar{\cup} D_2} = \mathrm{cl}_{D_1} \bar{\cup} \mathrm{cl}_{D_2},$$

and the converse is given below.

Proposition 7: If $\mathrm{cl} = \mathrm{cl}_1 \bar{\cup} \mathrm{cl}_2$, then $\mathrm{cl}_1 = \mathrm{cl}/_{V_1}$ and $\mathrm{cl}_2 = \mathrm{cl}/_{V_2}$. Moreover, if D is a loopless graph, then $\mathrm{cl}_D = \mathrm{cl}_1 \bar{\cup} \mathrm{cl}_2$ if and only if $\mathrm{cl}_1 = \mathrm{cl}_{D[V_1]}$, $\mathrm{cl}_2 = \mathrm{cl}_{D[V_2]}$, and $D = D[V_1] \bar{\cup} D[V_2]$.

Proof: The first claim is easy to prove. For the second claim, if $\mathrm{cl}_D = \mathrm{cl}_1 \bar{\cup} \mathrm{cl}_2$, then $\mathrm{cl}_1 = \mathrm{cl}_{D[V_1]}$ and $\mathrm{cl}_2 = \mathrm{cl}_{D[V_2]}$. Suppose the arc (v_1, v_2) is missing between V_1 and V_2. Then $\mathrm{cl}_D(V\backslash\{v_1, v_2\}) = V$ (since $\{v_1, v_2\}$ is acyclic), while $\mathrm{cl}_D/_{V_2} \bar{\cup} \mathrm{cl}_D/_{V_1}(V\backslash\{v_1, v_2\}) = V\backslash\{v_1, v_2\}$. The converse is trivial. ∎

VI. Guessing number and solvability graph

A. Definition and main results

The solvability graph extends the definition of the so-called guessing graph to all closure operators. Most of this section naturally extends [20]. Therefore, we shall omit certain proofs which are very similar to their counterparts in [20].

First of all, we need the counterpart of the guessing number of a graph for closure operators. Any partition f_i of A^r into at most $|A|$ parts is henceforth denoted as $f_i = \{P_a(f_i) : a \in A\}$, where some parts $P_a(f_i)$ are possibly empty. By extension, for any tuple $f = (f_1, \ldots, f_n)$, the partition f_V is denoted as $f_V = \{P_x(f_V) : x \in A^n\}$, where $P_x(f_V) = \bigcap_{i=1}^n P_{x_i}(f_i)$. We denote the set of words of A^n indexing non-empty parts of f_V as the image of f:

$$\mathrm{Im}(f) := \{x \in A^n : P_x(f_V) \neq \emptyset\}.$$

Definition 8: The guessing number of cl over A is given by

$$g(\mathrm{cl}, A) := \max_f \{\log_{|A|} |\mathrm{Im}(f)|\},$$

where the maximum is taken over all coding functions f for cl over A.

A coding function has an image of size $|A|^r$ if and only if it is a solution; therefore, cl is solvable over A if and only if $g(\mathrm{cl}, A) = r$.

Next, we define the solvability graph of closure operators.

Definition 9: The solvability graph $\mathrm{G}(\mathrm{cl}, A)$ has vertex set A^n and two words $x, y \in A^n$ are adjacent if and only if there exists no coding function f for cl over A such that $x, y \in \mathrm{Im}(f)$.

Proposition 8 below enumerates some properties of the solvability graph. In particular, Property 2 provides a concrete and elementary description of the edge set which makes adjacency between two configurations easily decidable.

Proposition 8: The solvability graph $\mathrm{G}(\mathrm{cl}, A)$ satisfies the following properties:

1) It has $|A|^n$ vertices.
2) Its edge set is $E = \bigcup_{S \subseteq V, v \in \mathrm{cl}(S)} E_{v,S}$, where $E_{v,S} = \{xy : x_S = y_S, x_v \neq y_v\}$.
3) It is vertex-transitive.

Proof: Property 1 follows from the definition. We now prove Property 2. f is a coding function if and only if $f_{S \cup v} = f_S$ for all $S \subseteq V$ and any $v \in \mathrm{cl}(S)$, which in turn is equivalent to $P_{x_{S \cup v}}(f_{S \cup v}) = P_{x_S}(f_S)$ for all $x \in \mathrm{Im}(f)$. Therefore, if $x_S = y_S$, $v \in \mathrm{cl}(S)$ and f is a coding function, we have $P_{x_S}(f_{x_S}) \subseteq P_{x_v}(f_{x_v})$ and $P_{x_S}(f_{x_S}) \subseteq P_{y_v}(f_{y_v})$, or in other words $x_v = y_v$.

Conversely, for distinct $xy \notin E$ we construct the following coding function. Let g be a partition of A^r into two nonempty parts and for all $v \in V$, let f_v have two parts $P_{x_v} = P_1(g)$ and $P_{y_v} = P_2(g)$ if $x_v \neq y_v$ and f_v have one part otherwise. Then, $\mathrm{Im}(f) = \{x, y\}$; for all $T \subseteq V$, $P_{x_T}(f_T) = P_{y_T}(f_T)$ if and only if $x_T = y_T$; and $f_V = g$. We now check that f is indeed a coding function: let $S \subseteq V$ and $v \in \mathrm{cl}(S)$. If $x_S = y_S$, then $x_v = y_v$ and hence $f_S = f_{S \cup v}$ has one part. Otherwise $f_S = g = f_V = f_{S \cup v}$.

For Property 3, we remark that $\mathrm{G}(\mathrm{cl}, A)$ is a Cayley graph [31], hence it is vertex-transitive. More explicitly, if we let

$A = \mathbb{Z}_{|A|}$, then $\phi(z) = z - x + y$ is an automorphism of the solvability graph which takes x to y for any $x, y \in A^n$. ∎

Corollary 1: The solvability graph for the uniform matroid $U_{r,n}$ has edge set $E = \{xy : d_H(x, y) \leq n - r\}$.

The main reason to study the solvability graph is given in Theorem 5 below.

Theorem 5: A set of words in A^n is an independent set of $\mathrm{G}(\mathrm{cl}, A)$ if and only if they are the image of A^r by a coding function for cl over A.

Proof: By definition of the solvability graph, the image of a coding function forms an independent set. Conversely, let $\{x^i\}_{i=1}^k$ be an independent set of the solvability graph $\mathrm{G}(\mathrm{cl}, A)$. Let g be a partition of A^r into k nonempty parts and let

$$P_{x_v^i}(f_v) := \bigcup_{j:x_v^j = x_v^i} P_j(g).$$

Then we have $\mathrm{Im}(f) = \{x^i\}_{i=1}^k$; for all $T \subseteq V$, $P_{x_T^i} = P_{x_T^j}$ if and only if $x_T^i = x_T^j$; and $f_V = g$. We now justify that f is a coding function. Let $v \in \mathrm{cl}(S)$, then for all i, $S_i := \{j : x_S^j = x_S^i\} = \{j : x_{S\cup v}^j = x_{S\cup v}^i\}$ and hence

$$P_{x_{S\cup v}^i} = \bigcup_{j \in S_i} P_j(g) = P_{x_S^i},$$

which means $f_S = f_{S\cup v}$. ∎

Corollary 2: We have $\log_{|A|} \alpha(\mathrm{G}(\mathrm{cl}, A)) = g(\mathrm{cl}, A)$ and hence $\alpha(\mathrm{G}(\mathrm{cl}, A)) = |A|^r$ if and only if cl is solvable over A.

In [20], we remark that the index coding problem asks for the chromatic number of the guessing graph of a digraph. We can extend the index coding problem to any closure operator and we say that cl is index-solvable over A if $b(\mathrm{cl}, A) := \log_{|A|} \chi(\mathrm{G}(\mathrm{cl}, A)) = n - r$. We have

$$g(\mathrm{cl}, A) + b(\mathrm{cl}, A) \geq n,$$
$$\lim_{|A|\to\infty} g(\mathrm{cl}, A) + \lim_{|A|\to\infty} b(\mathrm{cl}, A) = n$$

by (1). Therefore, although determining $g(\mathrm{cl}, A)$ and $b(\mathrm{cl}, A)$ are distinct over a fixed alphabet A, they are asymptotically equivalent. More strikingly, solvability and index-solvability are equivalent for finite alphabets too, as seen below.

Theorem 6: The closure operator cl is solvable over A if and only if it is index-solvable over A.

Proof: Let $\{x^i\}$ be an independent set of G and b be a basis of cl. Without loss, let $b = \{1, \ldots, r\}$). First, we remark that $x_b^i \neq x_b^j$ for all $i \neq j$, for otherwise $x_{V\setminus b}^i \neq x_{V\setminus b}^j$ and $x_b^i = x_b^j$ means that $x^i \sim x^j$. Secondly, let $A = \mathbb{Z}_{|A|}$, then for any $w \in A^{n-r}$ and any i, denote $x^i + w = (x_b^i, x_{V\setminus b}^i + w)$. Then it is easily shown that $S_w = \{x^i + w\}$ forms an independent set and that the family $\{S_w\}$ forms a partition of A^n into $|A|^{n-r}$ independent sets.

Conversely, if $\chi(G(\mathrm{cl}, A)) = |A|^{n-r}$, then $\alpha(G(\mathrm{cl}, A)) = |A|^r$ by (1). ∎

B. Neighbourhood and girth

Note that the relation "having an arc from u to v" cannot be expressed in terms of the D-closure. Indeed, all acyclic graphs on n vertices, from the empty graph to an acyclic tournament, all have the same closure operator $U_{0,n}$. However, the D-closure of the in-neighbourhood of a vertex can be described by means of the digraph closure.

Lemma 2: For any v and any $X \subseteq V\setminus\{v\}$, $v \in \mathrm{cl}_D(X)$ if and only if $\mathrm{cl}_D(v^-) \subseteq \mathrm{cl}_D(X)$.

Proof: Suppose $v \in \mathrm{cl}_D(X)\setminus X$, then $Y = \mathrm{cl}_D(X)\setminus X$ induces an acyclic subgraph and $Y^- \subseteq \mathrm{cl}_D(X)$; in particular, $v^- \subseteq \mathrm{cl}_D(X)$. Since $v \in \mathrm{cl}_D(v^-)$, we easily obtain the converse. ∎

We remark that if there is a loop on v, then there exists no set $X \subseteq V\setminus\{v\}$ such that $v \in \mathrm{cl}_D(X)$. Note that v^- is not necessarily an inner basis of its own closure, for instance this is trivial in nonempty acyclic digraphs.

Based on our results about closure operators associated to digraphs, we can define some concepts to any closure operators which generalise those of digraphs.

Definition 10: For any vertex v, the *degree* of v is

$$d_v := \min\{|X| : v \in \mathrm{cl}(X)\setminus X\}$$

if there exists such set X, or by convention is equal to 0 otherwise. We denote the minimum degree as δ.

Note that the degree (according to the closure operator cl_D) of a vertex of the digraph D is not necessarily equal to the size of its in-neighbourhood.

Definition 11: We say a subset X of vertices is *acyclic* if $\mathrm{cl}(V\setminus X) = V$. The *girth* γ of the closure operator as the minimum size of a non-acyclic subset of vertices.

Here, the girth of a digraph is equal to the girth of its closure operator.

We denote the maximum cardinality of a code over A of length n and minimum distance d as $M_A(n, d)$.

Proposition 9: For any cl, we have

$$\log_{|A|} M_A(n, n - \delta + 1) \leq g(\mathrm{cl}, A) \leq \log_{|A|} M_A(n, \gamma).$$

Since $\delta \leq r$ and $\gamma \leq n - r + 1$, we have $\gamma = n - \delta + 1$ if and only if $\mathrm{cl} = U_{r,n}$.

C. Combining closure operators

Recall the definitions of unions of closure operators in Section V. The following theorem is the counterpart of Propositions 6, 7, and 8 in [20].

Theorem 7: For any cl_1 and cl_2 defined on disjoint sets V_1 and V_2 of cardinalities n_1 and n_2, we have

$$G(\mathrm{cl}_1 \cup \mathrm{cl}_2, A) = G(\mathrm{cl}_1, A) \oplus G(\mathrm{cl}_2, A)$$
$$G(\mathrm{cl}_1 \vec{\cup} \mathrm{cl}_2, A) = G(\mathrm{cl}_1, A) \cdot G(\mathrm{cl}_2, A)$$
$$G(\mathrm{cl}_1 \bar{\cup} \mathrm{cl}_2, A) = G(\mathrm{cl}_1, A) \square G(\mathrm{cl}_2, A).$$

Therefore,

$$g(\mathrm{cl}_1 \cup \mathrm{cl}_2, A) = g(\mathrm{cl}_1 \vec{\cup} \mathrm{cl}_2, A) = g(\mathrm{cl}_1, A) + g(\mathrm{cl}_2, A)$$
$$b(\mathrm{cl}_1 \bar{\cup} \mathrm{cl}_2, A) = \max\{b(\mathrm{cl}_1, A), b(\mathrm{cl}_2, A)\}$$
$$g(\mathrm{cl}_1 \bar{\cup} \mathrm{cl}_2, A) \leq \min\{g(\mathrm{cl}_1, A) + n_2, g(\mathrm{cl}_2, A) + n_1\}.$$

Corollary 3: The following are equivalent:

- cl_1 and cl_2 are solvable over A
- $\mathrm{cl}_1 \cup \mathrm{cl}_2$ is solvable over A
- $\mathrm{cl}_1 \vec{\cup} \mathrm{cl}_2$ is solvable over A.

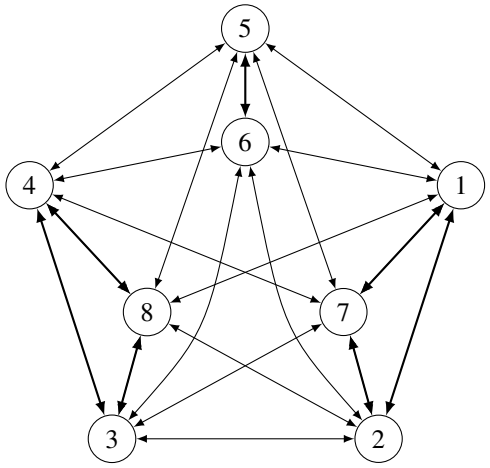


Fig. 3. The bidirectional union $E_3 \bar{\cup} \bar{C}_5$. The vertices of $\bar{C}_5$ form a basis; the highlighted disjoint cliques $127, 248, 56$ show that it is solvable.

Therefore, when studying solvability, we can only consider connected closure operators.

Corollary 4: Without loss, suppose $n_1 - r_1 \geq n_2 - r_2$, then we have the following list of properties each implying the next:

- cl_1 and cl_2 are solvable over A;
- cl_1 is solvable over A and $b(\mathrm{cl}_2, A) \leq n_1 - r_1$;
- $\mathrm{cl}_1 \bar{\cup} \mathrm{cl}_2$ is solvable over A;
- cl_1 is solvable over A and $g(\mathrm{cl}_2, A) \geq n_2 - n_1 + r_1$.

In particular, if $n_1 - r_1 = n_2 - r_2$, then $\mathrm{cl}_1 \bar{\cup} \mathrm{cl}_2$ is solvable over A if and only if cl_1 and cl_2 are solvable over A.

An example where cl_2 is not solvable, yet $\mathrm{cl}_1 \bar{\cup} \mathrm{cl}_2$ is solvable, is given in Figure 3.

The results on the bidirectional union can be viewed as "network sharing," illustrated in Figure 4. Suppose we have two solvable networks N_1 and N_2, where N_1 has the same number of or more intermediate nodes than N_2. Then N_2 can be plugged in to N_1, which can share its links with N_2 without compromising its solvability. In the resulting shared network, not only each source-destination pair of N_2 is there, but also each intermediate node yields an additional source-destination pair. As a result, the only intermediate nodes are those coming from N_1.

D. Combining alphabets

Let $[k] = \{1, \ldots, k\}$ for any positive integer k. We define a closure operator on $V \times [k]$ as follows. For any $v \in V$, let $[v] = \{(v, i) : i \in [k]\}$ and for any $X \subseteq V \times [k]$, denote $X_V = \{v \in V : [v] \subseteq X\}$. Then

$$\mathrm{cl}^{[k]}(X) := X \cup \{[v] : v \in \mathrm{cl}(X_V)\}.$$

This closure operator can be intuitively explained as follows. Consider the solvability problem of cl over the alphabet A^k. Each element of A^k is a vector of length k over A, then $\mathrm{cl}^{[k]}$ associates k according vertices $[v]$ to each $v \in V$, each new vertex (v, i) corresponding to the coordinate i. If $v \in \mathrm{cl}(Y)$ for some $Y \subseteq V$, then the local function f_v depends on f_Y. We can view $f_v : A^{kr} \to A^k$ (and hence all its coordinate functions) as depending on all coordinates of all vertices in Y, hence the definition of the closure operator.

In particular, for D construct $D^{[k]}$ as follows: its vertex set is $V \times [k]$ and its edge set is $\{((u, i), (v, j)) : (u, v) \in E(D)\}$. Then it is easy to check that $\mathrm{cl}_D^{[k]} = \mathrm{cl}_{D^{[k]}}$.

Proposition 10: We have the following properties:

1) $r(\mathrm{cl}^{[k]}) = kr(\mathrm{cl})$.
2) $G(\mathrm{cl}^{[k]}, A) \cong G(\mathrm{cl}, A^k)$ and hence $H(\mathrm{cl}^{[k]}, A) = kH(\mathrm{cl}, A^k)$.
3) If cl is connected, then so is $\mathrm{cl}^{[k]}$ for all k.

Proof: The proof of the first two claims is similar to that of [20, Proposition 10]. We now prove the last claim. For any $S \subseteq V \times [k]$, we denote $\lfloor S \rfloor = \bigcup_{v \in S_V} [v]$, $T = (V \times [k]) \backslash S$, and $\lceil T \rceil = T \cup (S \backslash \lfloor S \rfloor) = (V \times [k]) \backslash \lfloor S \rfloor$. Note that $S_V = \lfloor S \rfloor_V = V \backslash \lceil T \rceil_V$. Then we claim that if $\mathrm{cl}^{[k]}|_{\backslash T} = \mathrm{cl}^{[k]}|^{/T}$, then $\mathrm{cl}^{[k]}|_{\backslash \lceil T \rceil} = \mathrm{cl}^{[k]}|^{/\lceil T \rceil}$. For any $Y \subseteq \lfloor S \rfloor$, let $X = Y \cup (S \backslash \lfloor S \rfloor)$; then $X_V = Y_V$ and $X \cup T = Y \cup \lceil T \rceil$. We then have

$$\begin{aligned} \{[v] : v \in \mathrm{cl}(Y_V)\} \cap S &= \{[v] : v \in \mathrm{cl}(X_V)\} \cap S \\ &= \{[v] : v \in \mathrm{cl}((X \cup T)_V)\} \cap S \\ &= \{[v] : v \in \mathrm{cl}((Y \cup T)_V)\} \cap S, \end{aligned}$$

and in particular, then intersections with $\lfloor S \rfloor$ are equal, thus proving the claim.

Now suppose $\mathrm{cl}^{[k]}$ is disconnected, then $\mathrm{cl}^{[k]}|_{\backslash T} = \mathrm{cl}^{[k]}|^{/T}$ for some $T = \lceil T \rceil$ (and hence $S = \lfloor S \rfloor$ and $V = S_V \cup T_V$). Then for any $X \subseteq S$, $(X \cup T)_V = X_V \cup T_V$ and we have

$$\begin{aligned} \{[v] : v \in \mathrm{cl}(X_V) \cap S_V\} &= \{[v] : v \in \mathrm{cl}(X_V)\} \cap S \\ &= \{[v] : v \in \mathrm{cl}(X_V \cup T_V)\} \cap S \\ &= \{[v] : v \in \mathrm{cl}(X_V \cup T_V) \cap S_V\}, \end{aligned}$$

and hence $\mathrm{cl}|_{\backslash T_V}(X_V) = \mathrm{cl}|^{/T_V}(X_V)$ for all $X_V \subseteq S_V$. ∎

VII. Acknowledgment

The author would like to thank the team from Queen Mary, University of London for fruitful discussions and the anonymous reviewers for their valuable suggestions.

Appendix

A. Proof of Theorem 4

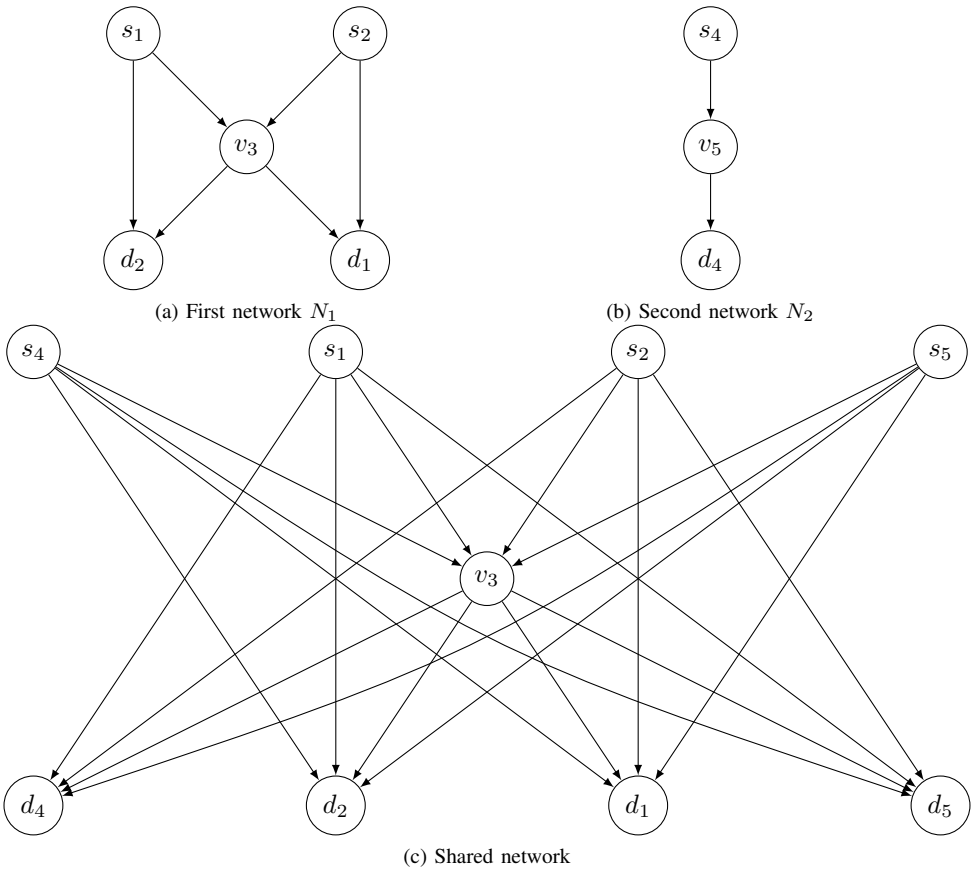
First of all, we justify why we only search for useless vertices in $T(D)$.

Lemma 3: If V_2 is useless, then $V_2 \subseteq T(D)$.

Proof: Let V_2 be a useless set. First of all, if X induces a chordless cycle, then it cannot entirely lie in V_2, for V_2 is acyclic. Suppose X does not lie entirely in V_1 either. Since X_1 is acyclic, we have $X_1 \subseteq \mathrm{cl}_D/_{V_2}(Y) \subseteq \mathrm{cl}_D(Y)$, where $Y = V_1 \backslash X_1$. Therefore, $X_2 \subseteq X^- \subseteq \mathrm{cl}_D(Y)$; gathering, we obtain $X \subseteq \mathrm{cl}_D(Y)$. More precisely, $X \subseteq \mathrm{cl}_D(Y) \backslash Y$ and hence X is acyclic, which is a contradiction. ∎

The following results ensure that we can remove useless vertices one by one.

Lemma 4: Let V_2 be useless in D and $v \in V_2$. Once v is removed from D, $V_2 \backslash v$ is useless in $D[V \backslash v]$.



(a) First network N_1

(b) Second network N_2

(c) Shared network

Fig. 4. Example of network sharing

Proof: $V_2\backslash v$ is clearly acyclic. For any $X \subseteq (V\backslash v)$, we have

$$\begin{aligned}\mathrm{cl}_{D[V\backslash v]}(X)\backslash(V_2\backslash v) &= \mathrm{cl}_D(X\cup v)\backslash V_2\\ &= \mathrm{cl}_D(X\cup V_2)\backslash V_2\\ &= (\mathrm{cl}_D(X\cup V_2)\backslash v)\backslash(V_2\backslash v)\\ &= \mathrm{cl}_{D[V\backslash v]}(X\cup(V_2\backslash v))\backslash(V_2\backslash v).\end{aligned}$$

■

Lemma 5: Let V_2 be useless in D and v the last vertex of V_2 according to an acyclic ordering (i.e., $v^+ \subseteq S$). Then $\{v\}$ is a useless set.

Proof: We only need to prove that $\{v\}$ is weak, i.e. for all $X \subseteq V\backslash v$, $\mathrm{cl}_D(X\cup v)\backslash v = \mathrm{cl}_D(X)\backslash v$. This clearly holds if $v \in \mathrm{cl}_D(X)$, hence let us assume that $v \notin \mathrm{cl}_D(X)$.

It is easy to show by induction on j that $c_D^j(X)\backslash X = c_D^j(X\cup v)\backslash(X\cup v)$ if and only if $c_D^j(X\cup v)\backslash(X\cup v)\cap v^+ = \emptyset$; in particular, $\mathrm{cl}_D(X)\backslash X = \mathrm{cl}_D(X\cup v)\backslash(X\cup v)$ if and only if $(\mathrm{cl}_D(X\cup v)\backslash(X\cup v))\cap v^+ = \emptyset$.

We have $v^+\cap(\mathrm{cl}_D(X)\backslash X) = \emptyset$. Since V_2 is weak, $\mathrm{cl}_D(X)\cap V_1 = \mathrm{cl}_D(X\cup v)\cap V_1$. Moreover, since $v^+ \subseteq V_1$, we have $\mathrm{cl}_D(X)\cap v^+ = \mathrm{cl}_D(X\cup v)\cap v^+$. Combining, we obtain $(\mathrm{cl}_D(X\cup v)\backslash(X\cup v))\cap v^+ = \emptyset$ and by the paragraph above, $\mathrm{cl}_D(X)\backslash X = \mathrm{cl}_D(X\cup v)\backslash(X\cup v)$, which yields $\mathrm{cl}_D(X)\backslash v = \mathrm{cl}_D(X\cup v)\backslash v$. ■

Next, we indicate an efficient way to check that a singleton is useless.

Lemma 6: For any vertex v, $\{v\}$ is useless if and only if for any $u \in v^+$, $v \in \mathrm{cl}_D(u^-\backslash v)$.

Proof: Suppose there exists $u \in v^+$ such that $v \notin \mathrm{cl}_D(u^-\backslash v)$. There is an edge from v to u, hence $u \notin \mathrm{cl}_D(u^-\backslash v)\backslash(u^-\backslash v)$ and $u \notin \mathrm{cl}_D(u^-\backslash v)\backslash u^-$. Since $u \in \mathrm{cl}_D(u^-)\backslash u^-$, we obtain $\mathrm{cl}_D(u^-)\backslash v \neq \mathrm{cl}_D(u^-\backslash v)\backslash v$ and $\{v\}$ is not weak.

Otherwise, suppose there exists X such that $\mathrm{cl}_D(X)\backslash v \neq \mathrm{cl}_D(X\cup v)\backslash v$; clearly $v \notin \mathrm{cl}_D(X)$. It is easy to show by induction on j that $c_D^j(X)\backslash X = c_D^j(X\cup v)\backslash(X\cup v)$ if and only if $\left(c_D^j(X\cup v)\backslash(X\cup v)\right)\cap v^+ = \emptyset$. Let $i = \min\{j : c_D^j(X)\backslash X \neq c_D^j(X\cup v)\backslash(X\cup v)\}$, then there exists $u \in (c_D^i(X\cup v)\backslash c_D^{i-1}(X\cup v))\cap v^+$. We have $u^-\backslash v \subseteq c_D^{i-1}(X)\mathrm{cl}_D(X)$, and hence $v \in \mathrm{cl}_D(u^-\backslash v) \subseteq \mathrm{cl}_D(X)$, which is the desired contradiction. ■

We can now prove the correctness of Algorithm 1. Clearly, the running time is polynomial.

Proof: First of all, Lemma 3 guarantees that the set of useless vertices lies in $T(D)$. At every iteration of the Repeat loop, if there exists a set of useless vertices in the new graph, then there exists a singleton $\{v\}$ which is useless by 5. By Lemma 6, the algorithm will find a useless vertex v if there exists one. Lemma 4 guarantees that after all the iterations, all the useless vertices will be removed. ■

REFERENCES

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4,

pp. 1204–1216, July 2000.
[2] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Transactions on Information Theory*, vol. 49, no. 2, pp. 371–381, February 2003.
[3] S. Riis, "Linear versus non-linear boolean functions in network flow," in *Proc. CISS*, Princeton, NJ, March 2004.
[4] R. Dougherty, C. Freiling, and K. Zeger, "Insufficiency of linear coding in network information flow," *IEEE Transactions on Information Theory*, vol. 51, no. 8, pp. 2745–2759, August 2005.
[5] R. Kötter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Transactions on Networking*, vol. 11, no. 5, pp. 782–795, October 2003.
[6] T. Ho, M. Médard, R. Kötter, D. R. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4413–4430, October 2006.
[7] R. Dougherty, C. Freiling, and K. Zeger, "Linearity and solvability in multicast networks," *IEEE Transactions on Information Theory*, vol. 50, no. 10, pp. 2243–2256, October 2004.
[8] ——, "Unachievability of network coding capacity," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2365–2372, June 2006.
[9] ——, "Networks, matroids, and non-Shannon information inequalities," *IEEE Transactions on Information Theory*, vol. 53, no. 6, pp. 1949–1969, June 2007.
[10] T. Chan and A. Grant, "Dualities between entropy functions and network codes," *IEEE Transactions on Information Theory*, vol. 54, no. 10, pp. 4470–4487, October 2008.
[11] ——, "On capacity regions of non-multicast networks," in *Proc. ISIT*, Austin, TX, 2010, pp. 2378–2382.
[12] T. H. Chan, "On the optimality of group network codes," in *Proc. ISIT*, Adelaide, Australia, 2005, pp. 1992–1996.
[13] S. Riis, "Utilising public information in network coding," in *General Theory of Information Transfer and Combinatorics*, 2006, pp. 866–897.
[14] R. Dougherty and K. Zeger, "Nonreversibility and equivalent constructions of multiple unicast networks," *IEEE Transactions on Information Theory*, vol. 52, no. 11, pp. 1287–1291, November 2006.
[15] S. Riis, "Information flows, graphs and their guessing numbers," *The Electronic Journal of Combinatorics*, vol. 14, pp. 1–17, 2007.
[16] T. Wu, P. Cameron, and S. Riis, "On the guessing number of shift graphs," *Journal of Discrete Algorithms*, vol. 7, pp. 220–226, 2009.
[17] S. Riis, "Reversible and irreversible information networks," *IEEE Transactions on Information Theory*, vol. 53, no. 11, pp. 4339–4349, November 2007.
[18] T. Chan, "Recent progresses in characterising information inequalities," *Entropy*, vol. 13, pp. 379–401, 2011.
[19] D. Christofides and K. Markström, "The guessing number of undirected graphs," *Electronic Journal of Combinatorics*, vol. 18, no. 1, pp. 1–19, 2011.
[20] M. Gadouleau and S. Riis, "Graph-theoretical constructions for graph entropy and network coding based communications," *IEEE Transactions on Information Theory*, vol. 57, no. 10, pp. 6703–6717, October 2011.
[21] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, p. 61213, November 1979.
[22] E. F. Brickell and D. M. Davenport, "On the classification of ideal secret sharing schemes," *J. Cryptology*, vol. 4, pp. 123–134, 1991.
[23] P. D. Seymour, "On secret-sharing matroids," *J. Combin. Theory Ser. B*, vol. 56, pp. 69–73, 1992.
[24] F. Matúš, "Matroid representations by partitions," *Discrete Math.*, vol. 203, pp. 169–194, 1999.
[25] P. J. Cameron, M. Gadouleau, and S. Riis, "Combinatorial representations," *Journal of Combinatorial Theory, Series A*, vol. 120, no. 3, pp. 671–682, April 2013.
[26] C.-C. Wang and N. B. Shroff, "Pairwise intersession network coding on directed networks," *IEEE Transactions on Information Theory*, vol. 56, no. 8, pp. 38 793–3900, August 2010.
[27] G. Birkhoff, *Lattice theory*. American Mathematical Society Colloquium Publications, 1948.
[28] J. G. Oxley, *Matroid Theory*. Oxford University Press, 2006.
[29] R. A. Bailey, *Association Schemes: Designed Experiments, Algebra and Combinatorics*, ser. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge: Cambridge University Press, 2004.
[30] J. Bang-Jensen and G. Gutin, *Digraphs: Theory, Algorithms and Applications*. Springer, 2009.
[31] C. D. Godsil and G. Royle, *Algebraic Graph Theory*, ser. Graduate Texts in Mathematics. Springer-Verlag, 2001, vol. 207.
[32] L. Lovász, "On the ratio of optimal integral and fractional covers," *Discrete Mathematics*, vol. 13, pp. 383–390, 1975.
[33] R. Bose, S. Shrikhande, and E. Parker, "Further results on the construction of mutually orthogonal latin squares and the falsity of Euler's conjecture," *Canadian Journal of Mathematics*, vol. 12, pp. 189–203, 1960.

PLACE PHOTO HERE

Maximilien Gadouleau (S'06–M'10) received an equivalent to the M.S. in Electrical and Computer Engineering from Esigelec, Saint-Etienne du Rouvray, France, in 2004 and the M.S. and Ph.D. in Computer Engineering from Lehigh University, Bethlehem, PA, in 2005 and 2009, respectively.

From 2009 to 2010, he was a postdoctoral researcher at CReSTIC, Université de Reims Champagne-Ardenne, Reims, France. From 2010 to 2011, he held a postdoctoral research assitantship at the School of Electronic Engineering and Computer Science, Queen Mary, University of London. In 2012, he joined the School of Engineering and Computing Sciences at Durham Univerity, as a lecturer in Computer Science. His current research interests are coding theory, network coding, and cryptography and their relations to combinatorics, algebra, and graph theory.

Dr. Gadouleau is a member of the IEEE Information Theory Society.