

# GRAPHS INDUCED BY GRAY CODES

ELIZABETH L. WILMER AND MICHAEL D. ERNST

ABSTRACT. We disprove a conjecture of Bultena and Ruskey [1], that all trees which are cyclic graphs of cyclic Gray codes have diameter 2 or 4, by producing codes whose cyclic graphs are trees of arbitrarily large diameter. We answer affirmatively two other questions from [1], showing that strongly  $P_n \times P_n$ -compatible codes exist and that it is possible for a cyclic code to induce a cyclic digraph with no bidirectional edge.

A major tool in these proofs is our introduction of *supercomposite* Gray codes; these generalize the standard reflected Gray code by allowing shifts. We find supercomposite Gray codes which induce large diameter trees, but also show that many trees are not induced by supercomposite Gray codes.

We also find the first infinite family of connected graphs known not to be induced by any Gray code — trees of diameter 3 with no vertices of degree 2.

## 1. INTRODUCTION

An  $n$ -bit Gray code  $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_N)$ ,  $N = 2^n$ , lists all the binary  $n$ -tuples (“codewords”) so that consecutive  $n$ -tuples differ in one bit. In a *cyclic* code, the first and last  $n$ -tuples also differ in one bit. Gray codes can be viewed as Hamiltonian paths on the hypercube graph; cyclic codes correspond to Hamiltonian cycles. Two Gray codes are *isomorphic* when one is carried to the other by a hypercube isomorphism.

The *transition sequence*  $\tau(\mathbf{B}) = (\tau_1, \tau_2, \dots, \tau_{N-1})$  of an  $n$ -bit Gray code  $\mathbf{B}$  lists the bit positions  $\tau_i \in [n] = \{1, 2, \dots, n\}$  where  $\mathbf{b}_i$  and  $\mathbf{b}_{i+1}$  differ. When  $\mathbf{B}$  is cyclic, its *closing transition*  $\tau_N$  is the position where  $\mathbf{b}_N$  and  $\mathbf{b}_1$  differ. We say  $\tau$  *generates*  $\mathbf{B}$  when  $\tau = \tau(\mathbf{B})$ . As transition sequences can be characterized simply and determine codes up to isomorphism, we treat codes and sequences interchangeably.

**Proposition 1.1** (Gilbert [3]). *Let  $\tau = (\tau_1, \tau_2, \dots, \tau_{N-1})$ , where  $N = 2^n$ .*

- (1)  *$\tau$  generates an  $n$ -bit Gray code if and only if every contiguous subsequence  $\tau_k, \tau_{k+1}, \dots, \tau_{k+l}$  contains some element of  $[n]$  an odd number of times.*
- (2)  *$\tau$  generates a cyclic Gray code if and only if  $\tau$  generates a Gray code and exactly one element of  $[n]$  appears an odd number of times in  $\tau$ ; that element is the closing transition.*

The graph  $G_{\mathbf{B}}$  induced by the Gray code  $\mathbf{B}$  has vertex set  $[n]$  and edge set  $\{\{\tau_i, \tau_{i+1}\} : i \in [N-1]\}$ , where  $\tau(\mathbf{B}) = (\tau_1, \tau_2, \dots, \tau_{N-1})$ . The vertices of  $G_{\mathbf{B}}$  correspond to bit positions; vertices  $i$  and  $j$  are adjacent when bit positions  $i$  and  $j$  flip consecutively during the code  $\mathbf{B}$ . Clearly,  $\tau(\mathbf{B})$  determines  $G_{\mathbf{B}}$ . When  $\mathbf{B}$  is cyclic with closing transition  $\tau_N$ , its *cyclic graph*  $\overline{G}_{\mathbf{B}}$  is  $G_{\mathbf{B}}$  together with the edges  $\{\tau_{N-1}, \tau_N\}$  and  $\{\tau_N, \tau_1\}$  (which may or may not already appear in  $G_{\mathbf{B}}$ ). Proposition 1.1 ensures that  $\tau(\mathbf{B})$  determines  $\overline{G}_{\mathbf{B}}$  whenever  $\mathbf{B}$  is cyclic.

Given a graph  $G$  with vertex set  $[n]$ , we call an  $n$ -bit Gray code  $\mathbf{B}$  *G-compatible* when  $G_{\mathbf{B}}$  is a spanning subgraph of  $G$  — that is, when  $\tau(\mathbf{B})$  is a walk on the edges

$$\mathbf{P}_4 = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 1, & 1, & 1, & 1, & 1, & 1, & 1, & 1 \end{pmatrix}$$



FIGURE 1. The Gray code  $\mathbf{P}_4$  has  $\tau(\mathbf{P}_4) = (3, 1, 2, 1, 3, 1, 2, 4, 2, 1, 3, 1, 2, 1, 3)$  and closing transition 4.

of  $G$  that visits every vertex. When  $\mathbf{B}$  is a cyclic  $n$ -bit Gray code (as are most codes we consider), we distinguish another level of compatibility. A cyclic code  $\mathbf{B}$  is *strongly  $G$ -compatible* when  $\overline{G}_{\mathbf{B}}$  is a spanning subgraph of  $G$ —that is, when  $\tau_1, \tau_2, \dots, \tau_{N-1}, \tau_N, \tau_1$  is a closed walk on  $G$  visiting every vertex.

*Example.* We write  $P_n$  for the  $n$ -path,  $C_n$  for the  $n$ -cycle, and  $K_n$  for the  $n$ -vertex complete graph. Figure 1 shows  $\mathbf{P}_4$ , a cyclic 4-bit code, together with  $G_{\mathbf{P}_4} = P_4$  and  $\overline{G}_{\mathbf{P}_4} = C_4$ . While the code  $\mathbf{P}_4$  is  $P_4$ -compatible, it is not strongly  $P_4$ -compatible. The code  $\mathbf{P}_4$  is, however, strongly  $C_4$ -compatible and strongly  $K_4$ -compatible.

*Remark.* Because every bit must flip during a Gray code, every  $G$ -compatible code induces a connected spanning subgraph of  $G$ . When  $T$  is a tree, every  $T$ -compatible code induces  $T$  and every strongly  $T$ -compatible code has cyclic graph  $T$ .

Slater [7, 8] was the first to ask: for which graphs  $G$  do  $G$ -compatible Gray codes exist—or, even better, strongly  $G$ -compatible cyclic codes? Bultena and Ruskey [1] independently arrived at this question, motivated in part by the search for Hamiltonian cycles on the cube-connected-cycle graph allowing simple traversal of the processors of certain parallel computers. Many types of restricted Gray codes, often motivated by applications, have been studied—see [2, 4, 6] for surveys.

The current work makes progress in both positive (constructing Gray codes that induce new graphs) and negative (finding graphs  $G$  such that no  $G$ -compatible code exists) directions. Section 2 introduces *supercomposite* Gray codes. Bultena and Ruskey [1] conjecture that all trees induced by cyclic Gray codes have diameter 2 or 4; we disprove their conjecture by finding supercomposite codes whose cyclic graphs are trees of arbitrarily large diameter. We also find supercomposite Gray codes strongly compatible with non-degenerate grid graphs and show that many trees are not induced by supercomposite Gray codes. In Section 3, on *digraphs of Gray codes*, we construct a family of cyclic Gray codes whose cyclic digraphs contain no bidirectional edges. The concluding Section 4 surveys current understanding of which graphs are induced by Gray codes. We give the first infinite family of trees that can be proved to not be induced by codes, determine all 7-vertex graphs  $G$  for which  $G$ -compatible codes exist, and pose some new questions.

## 2. SUPERCOMPOSITE GRAY CODES

**2.1. Definitions.** Supercomposite Gray codes are all those that can be built by two simple operations: shifting and reflecting. The term “supercomposite” is inspired by Gilbert [3], who gave a similar definition of “ultracomposite” Gray codes.

Breaking a cyclic Gray code between any two consecutive codewords yields another Gray code. Define the  $k$ -th shift of a cyclic code  $\mathbf{B}$  to be

$$S^k(\mathbf{B}) = (\mathbf{b}_{k+1}, \mathbf{b}_{k+2}, \dots, \mathbf{b}_N, \mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_k).$$

(Shift parameters should be taken mod  $N$ . For example,  $S^0(\mathbf{B}) = \mathbf{B}$ .) When  $\mathbf{B}$  has transition sequence  $(\tau_1, \dots, \tau_{N-1})$  and closing transition  $\tau_N$ ,  $S^k(\mathbf{B})$  has closing transition  $\tau_k$  and transition sequence

$$(\tau_{k+1}, \tau_{k+2}, \dots, \tau_{N-1}, \tau_N, \tau_1, \tau_2, \dots, \tau_{k-1}).$$

Although  $\overline{G}(S^k(\mathbf{B})) = \overline{G}(\mathbf{B})$  for any  $k$ , shifts can add or subtract edges in the (ordinary) induced graph of a code. Define a cyclic Gray code  $\mathbf{B}$  to be *sufficient* when its graph is constant under shifts; that is,  $G_{S^k(\mathbf{B})} = G_{\mathbf{B}}$  for every  $k$ . A cyclic code  $\mathbf{B}$  with transition sequence  $(\tau_1, \dots, \tau_{N-1})$  is sufficient if and only if every edge in  $\overline{G}_{\mathbf{B}}$  has two non-consecutive appearances in the closed walk  $\tau_1, \tau_2, \dots, \tau_{N-1}, \tau_N, \tau_1$ ; thus,  $\mathbf{B}$  sufficient implies  $G_{\mathbf{B}} = \overline{G}_{\mathbf{B}}$ .

The *reflection* of an  $n$ -bit Gray code  $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_N)$  is the  $n+1$ -bit code

$$\text{Rf}(\mathbf{B}) = \begin{pmatrix} \mathbf{b}_1 & \mathbf{b}_2 & & \mathbf{b}_N & \mathbf{b}_N & & \mathbf{b}_2 & \mathbf{b}_1 \\ 0, & 0, & \dots, & 0, & 1, & \dots, & 1, & 1 \end{pmatrix}.$$

This code is cyclic with closing transition  $n+1$ . When  $\tau(\mathbf{B}) = (\tau_1, \tau_2, \dots, \tau_{N-1})$ ,

$$\tau(\text{Rf}(\mathbf{B})) = (\tau_1, \tau_2, \dots, \tau_{N-1}, n+1, \tau_{N-1}, \dots, \tau_2, \tau_1).$$

Reflecting a code simply adds a leaf to its graph. Shifting before reflection changes where the leaf is added; shifting after reflection can add a second edge. Proposition 2.1 anatomizes these effects.

**Proposition 2.1.** *Let  $\tau = (\tau_1, \tau_2, \dots, \tau_{N-1})$  generate a cyclic  $n$ -bit Gray code with closing transition  $\tau_N$  and let  $j$  and  $k$  be integers. Then:*

- (1)  $G_{\text{Rf}(S^j(\tau))}$  is  $G_{S^j(\tau)}$  plus the new leaf  $n+1$  attached to  $\tau_{j-1}$ . In  $\overline{G}_{\text{Rf}(S^j(\tau))}$ ,  $n+1$  is adjacent to both  $\tau_{j-1}$  and  $\tau_{j+1}$ .
- (2)  $G_{S^k(\text{Rf}(S^j(\tau)))}$  is  $G_{S^j(\tau)}$  plus the new vertex  $n+1$  attached to:  $\tau_{j-1}$  when  $k \equiv 0 \pmod{2N}$ ,  $\tau_{j+1}$  when  $k \equiv N \pmod{2N}$ , and both  $\tau_{j-1}$  and  $\tau_{j+1}$  otherwise.
- (3)  $\text{Rf}(S^j(\tau))$  is sufficient if and only if  $\tau_{j+1} = \tau_{j-1}$ .

*Proof.* (1) Because  $n+1$  appears once in  $\text{Rf}(S^j(\tau))$ , between two occurrences of  $\tau_{j-1}$ ,  $n+1$  is adjacent only to  $\tau_{j-1}$  in  $G_{\text{Rf}(S^j(\tau))}$ . As closing transition,  $n+1$  is flanked on both sides by  $\tau_{j+1}$ . Thus  $n+1$  is also adjacent to  $\tau_{j+1}$  in  $\overline{G}_{\text{Rf}(S^j(\tau))}$ .

(2) Every edge of  $\overline{G}_{\text{Rf}(S^j(\tau))}$  not incident to  $n+1$  appears on both sides of the reflection and thus is in  $G_{S^k(\text{Rf}(S^j(\tau)))}$  for every  $k$ . The edge  $\{n+1, \tau_{j+1}\}$  does not appear in  $G_{S^0(\text{Rf}(S^j(\tau)))} = G_{\text{Rf}(S^j(\tau))}$  unless  $\tau_{j+1} = \tau_{j-1}$ ; similarly, the edge  $\{n+1, \tau_{j-1}\}$  does not appear in  $G_{S^N(\text{Rf}(S^j(\tau)))}$  unless  $\tau_{j+1} = \tau_{j-1}$ .

(3) This follows immediately from (2). □

*Example.* Reflecting  $(0, 1)$   $n-1$  times yields the *standard reflected Gray code*  $\mathbf{R}_n$  (as featured in F. Gray's patent [5]). Repeatedly applying Proposition 2.1 gives  $G_{\mathbf{R}_n} = \overline{G}_{\mathbf{R}_n} = K_{1, n-1}$  (a star graph with one central vertex connected to  $n-1$  leaves), as noted in [1, 9]. These codes are sufficient.

*Remark.* Whenever  $\tau$  is sufficient,  $\tau_{j-1} \neq \tau_{j+1}$ , and  $k \not\equiv 0, N \pmod{2N}$ , the graph induced by  $S^k(\text{Rf}(S^j(\tau)))$  contains a 4-cycle with vertices  $\tau_{j-1}, \tau_j, \tau_{j+1}, n+1$ .

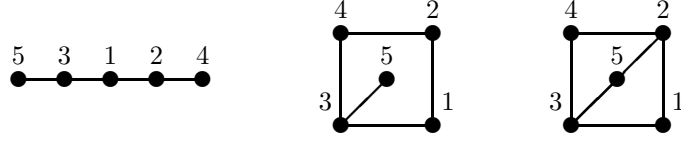


FIGURE 2. The graphs induced by  $\mathbf{P}_5 = \text{Rf}(\mathbf{P}_4)$ ,  $\text{Rf}(\text{S}^2(\mathbf{P}_4))$ , and  $\text{S}^4(\text{Rf}(\text{S}^2(\mathbf{P}_4)))$ .

*Example.* The insufficient code  $\mathbf{P}_4 = \text{Rf}(\text{S}^3(\mathbf{R}_3))$  shown in Figure 1 has transition sequence starting and ending with 3. The code  $\mathbf{P}_5 = \text{Rf}(\mathbf{P}_4)$  is thus sufficient; it, and all of its shifts, induce  $P_5$ . The code  $\text{Rf}(\text{S}^2(\mathbf{P}_4))$  is insufficient, with transition sequence  $(2, 1, 3, 1, 2, 4, 2, 1, 3, 1, 2, 1, 3, 4, 3, 5, 3, 4, 3, 1, 2, 1, 3, 1, 2, 4, 2, 1, 3, 1, 2)$  and closing transition 5. See Figure 2.

We define  $\mathcal{S}_n$ , the collection of *supercomposite*  $n$ -bit Gray codes, inductively. The members of  $\mathcal{S}_1$  are the 1-bit Gray codes  $(0, 1)$  and  $(1, 0)$ . For  $n > 1$ ,

$$\mathcal{S}_n = \{\text{S}^k(\text{Rf}(\mathbf{B})) \mid \mathbf{B} \in \mathcal{S}_{n-1}, k \in \mathbb{Z}\}.$$

That is,  $\mathcal{S}_n$  contains all cyclic shifts of reflections of codes in  $\mathcal{S}_{n-1}$ . In fact,  $\mathcal{S}_n$  contains all  $n$ -bit codes constructible from  $(0, 1)$  by shifts and reflections.

Proposition 2.1 has many consequences for the structure of graphs induced by supercomposite codes. All supercomposite Gray codes are cyclic. When  $\mathbf{B} \in \mathcal{S}_n$ ,  $n$  has degree at most 2 in  $G_{\mathbf{B}}$ . All codes in  $\mathcal{S}_3$  induce 2 edges; induction on the number of vertices shows that codes in  $\mathcal{S}_n$ ,  $n \geq 3$ , induce at most  $2n - 4$  edges. Furthermore, whenever  $n$  has degree 2 in a graph induced by a code in  $\mathcal{S}_n$ , its two neighbors have another common neighbor. Thus every graph induced by a supercomposite Gray code is two-colorable—just color each vertex as it is added.

**2.2. Trees induced by supercomposite Gray codes.** Let  $\mathcal{T}_n = \{\tau \in \mathcal{S}_n : G_{\tau} \text{ is a tree}\}$ . That is,  $\mathcal{T}_n$  contains all supercomposite codes inducing trees. Which trees are actually induced by codes in  $\mathcal{T}_n$ ? Shifting a cyclic code can enable the next vertex to be attached to any vertex of the current graph. However, among the supercomposites, only sufficient codes can be shifted arbitrarily without introducing cycles. Furthermore, as detailed in Proposition 2.2, attaching a new leaf to a leaf requires that the construction process include an insufficient code (the insufficiency can either precede or follow attaching the leaf to a leaf). These restrictions prevent many trees—most interestingly, paths with more than 6 vertices—from being induced by supercomposite codes.

**Proposition 2.2.** *Let  $\sigma \in \mathcal{T}_{n+1}$ ,  $n \geq 3$ , such that the neighbor in  $G_{\sigma}$  of  $n + 1$  is a vertex  $u$  with only one other neighbor,  $v$ . Then one of the following must occur:*

1.  $\sigma = \text{Rf}(\tau)$  or  $\sigma = \text{S}^N(\text{Rf}(\tau))$ , where  $\tau \in \mathcal{T}_n$  is sufficient; then  $\sigma$  is not sufficient, and  $\sigma_1 = \sigma_{2N-1} = w$ , where  $w$  is a neighbor of  $v$ .
2.  $\sigma = \text{S}^l(\text{Rf}(\text{Rf}(\rho)))$  or  $\sigma = \text{S}^l(\text{Rf}(\text{S}^{N/2}(\text{Rf}(\rho))))$ , where  $\rho \in \mathcal{T}_{n-1}$  is sufficient and  $l$  is an integer. Then  $\sigma$  is sufficient and, in  $G_{\sigma}$ ,  $n$  is a leaf adjacent to a neighbor  $w$  of  $v$ .

*Proof.* By the definition of  $\mathcal{T}_{n+1}$ ,  $\sigma = \text{S}^l(\text{Rf}(\tau))$  for some  $l$  and some  $\tau \in \mathcal{T}_n$  ( $G_{\tau}$  cannot contain cycles). By Proposition 2.1, the only neighbor of  $u$  in  $G_{\tau}$  is  $v$ .

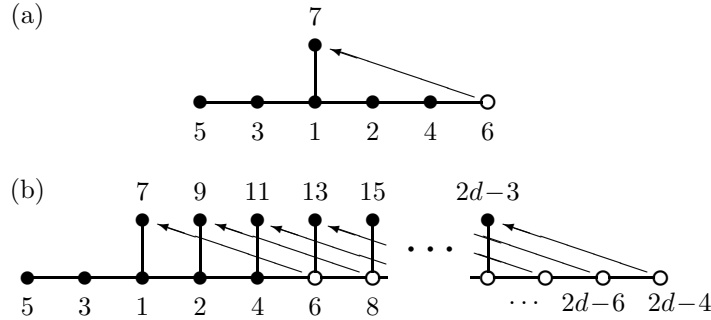


FIGURE 3. (a) Failing to find supercomposite Gray codes inducing long paths. (b) The graph induced by  $\mathbf{L}_d$ . White vertices determine the next vertex added, as indicated by the arrows.

First, we assume  $\tau$  is sufficient and consider the possible values of  $l$ . When  $l = 0$ , i.e.  $\sigma = \text{Rf}(\tau)$ , we must have  $\tau_{N-1} = u$  and  $\tau_{N-2} = \tau_0 = v$ . If  $\sigma$  were sufficient, Proposition 2.1(3) would imply  $\tau_1 = u$ . But then  $\tau_2 = \tau_0 = \tau_{N-2} = v$ , contradicting Proposition 1.1 (as long as  $\tau$  is a code on more than 2 bits). Thus,  $\sigma$  is not sufficient and  $\sigma_1 = \sigma_{2N-1} = \tau_1 = w$ , the other neighbor of  $v$ , as claimed.

When  $\sigma = \text{S}^N(\text{Rf}(\tau))$  with  $\tau$  sufficient, applying the same argument, with  $\tau_1$  and  $\tau_{N-1}$  switched, yields the same conclusions.

If it were the case that  $\sigma = \text{S}^l(\text{Rf}(\tau))$  with  $\tau$  sufficient and  $l \not\equiv 0, N \pmod{2N}$ , Proposition 2.1(2) would imply that  $n + 1$  is adjacent to both  $\tau_1$  and  $\tau_{N-1}$  in  $G_\sigma$ . But then  $\tau_1 = \tau_{N-1} = u$ , so  $\tau_{N-2} = \tau_0 = \tau_2 = v$ , and we obtain the same contradiction as before. Thus, whenever  $\tau$  is sufficient, the code falls into case (1).

What happens when  $\sigma = \text{S}^l(\text{Rf}(\tau))$ , but  $\tau$  is not sufficient? By Proposition 2.1(2),  $\tau \in \mathcal{T}_n$  and insufficient implies that  $\tau = \text{Rf}(\rho)$  or  $\tau = \text{S}^{N/2}(\text{Rf}(\rho))$  for some code  $\rho \in \mathcal{T}_{n-1}$  satisfying  $\rho_1 \neq \rho_{N/2-1}$ . By Proposition 2.1, the only way that we can have  $\rho_1 \neq \rho_{N/2-1}$  and  $\rho$  inducing a tree is if  $\rho$  is a non-trivial shift of a sufficient code (indeed, one on at least 3 bits). But then  $\rho_{N/2-1}, \rho_0, \rho_1$  is a walk in  $G_\rho$ , and  $G_\rho$  is a subgraph of  $G_\sigma$ . In the case that  $\tau = \text{Rf}(\rho)$ , we have  $\rho_1 = \tau_1 = \tau_{N-1} = u$ . By Proposition 2.1(3),  $\text{Rf}(\tau)$  is sufficient, so  $\sigma = \text{S}^l(\text{Rf}(\tau))$  is sufficient. Because  $\rho_0 = v$ ,  $\rho_{N/2-1}$  must be a neighbor  $w$  of  $v$ , and  $n$  is attached to  $w$  in  $G_\tau$ . Similarly, when  $\tau = \text{S}^{N/2}(\text{Rf}(\rho))$ ,  $\rho_{N/2-1} = \tau_1 = \tau_{N-1} = u$ ; again,  $\sigma$  is sufficient and  $\rho_0 = v$ , so  $\rho_1 = w$ , and  $n$  is attached to  $w$  in  $G_\tau$ . We have confirmed all claims in case (2).  $\square$

*Remark.* Proposition 2.2 says that when constructing supercomposite codes inducing trees, a leaf can only be attached to the leaf  $u$  if another “buddy” leaf is attached two vertices away (to a neighbor  $w$  of  $v$ ). In case (1),  $\sigma$  is insufficient. To proceed to a larger tree, we are forced either to reflect or to half-shift, then reflect. Either way, we get leaves adjacent to both  $u$  (as desired) and  $w$  (the forced “buddy”). In case (2), we have already added the “buddy” leaf to a neighbor  $w$  of  $v$ ; we then attach  $n + 1$  to  $u$ , and  $\sigma$  is sufficient — but its predecessor  $\tau$  was not.

*Example.* Can we build supercomposite Gray codes inducing paths? Because every subtree of a path is a path, all codes in the construction of such a code must also induce paths. Up to isomorphism,  $\mathbf{R}_2, \mathbf{R}_3, \mathbf{P}_4$  and  $\mathbf{P}_5$  are the only supercomposite Gray codes inducing  $P_2, P_3, P_4$ , and  $P_5$ , respectively. The only  $k$  for which

$\text{Rf}(S^k(\mathbf{P}_5))$  adds leaves to the ends of  $G_{\mathbf{P}_5}$  are 1 or 17 (6 attached to 5) or 9 or 25 (6 attached to 4). See Figure 3(a). Because  $\mathbf{P}_5$  is sufficient, shifting and reflecting to add a 6 to either 4 or 5 will put us into case (1) of Proposition 2.2. The next leaf must be attached to the center 1 of  $P_5$ . Thus no supercomposite Gray code induces  $P_7$  or any longer path.

We can continue the above example, repeatedly shifting and reflecting to add a leaf at the end of the longest path, then accepting the reflection forced by case (1) of Proposition 2.2, to produce cyclic Gray codes inducing trees of increasing diameter. More specifically, let  $\mathbf{L}_4 = \mathbf{P}_5$ ,  $\mathbf{L}_5 = \text{Rf}(\text{Rf}(S^9(\mathbf{L}_4)))$ , and

$$\mathbf{L}_d = \text{Rf}(\text{Rf}(S^{2^{2d-7}+1}(\mathbf{L}_{d-1})))$$

for  $d > 5$ . Then  $\mathbf{L}_d$  is a sufficient supercomposite code whose cyclic graph is a tree of diameter  $d$ . At each stage, the shift and the first reflection attach  $2d - 4$  to the leaf  $2d - 6$ ; the second reflection adds the ‘‘buddy’’ leaf  $2d - 3$  three steps back in the path and yields a sufficient code, allowing us to shift without introducing cycles at the next stage. See Figure 3(b). This construction disproves the conjecture of Bultena and Ruskey [1] that all trees that are cyclic graphs of cyclic codes have diameter 2 or 4. We have shown:

**Theorem 2.3.** *There exist trees of arbitrarily large diameter induced by sufficient supercomposite Gray codes.*

Bultena and Ruskey [1] also ask whether strongly  $P_n \times P_n$ -compatible codes exist. Paths don’t seem to have enough edges to allow Gray codes using only those edges; do square grids have enough edges? Yes. In fact, we can use the  $\mathbf{L}_d$ ’s to build strongly grid-compatible codes. First we must show that given an interior vertex  $v$  of a graph induced by a sufficient code in  $\mathcal{T}_n$ , it’s always possible to shift and reflect so that a new leaf is attached to  $v$ , while sufficiency is maintained. Thus we will be able to add as many leaves as we like to arbitrary interior vertices.

**Proposition 2.4.** *Let  $\tau \in \mathcal{T}_n$ ,  $n \geq 3$ , be sufficient. For any interior vertex  $v$  of  $G_\tau$ , there exists a  $k$  such that  $\text{Rf}(S^k(\tau))$  is a sufficient code with  $G_\tau$ , plus a new leaf adjacent to  $v$ , as its cyclic graph.*

*Proof.* By Proposition 2.1, it will suffice to find a  $k$  such that  $\tau_{k-1} = \tau_{k+1} = v$ . In fact, we prove the broader claim that for any  $\tau \in \mathcal{T}_n$  (sufficient or not) and any interior vertex  $v$  of  $G_\tau$ , there exists a  $k$ ,  $1 < k < N - 1$ , such that  $\tau_{k-1} = \tau_{k+1} = v$ . We use induction on  $n$ . As base case, note that every element of  $\mathcal{S}_3$  is isomorphic to a shift of the sufficient code  $\mathbf{R}_3 = (1, 2, 1, 3, 1, 2, 1)$ , for which our claim clearly holds. Now, fix  $n \geq 3$  and assume that for any  $\tau \in \mathcal{T}_n$  and any vertex  $v$  of degree at least 2 in  $G_\tau$ , there exists a  $k$ ,  $1 < k < N$ , such that  $\tau_{k-1} = \tau_{k+1} = v$ . Let  $\sigma = (\sigma_1, \dots, \sigma_{2N-1}) \in \mathcal{T}_{n+1}$ . Then  $\sigma = S^j(\text{Rf}(\tau))$  for some  $\tau \in \mathcal{S}_n$  and some  $j$ .

If  $\sigma = S^0(\text{Rf}(\tau)) = \text{Rf}(\tau)$ , then Proposition 2.1 implies that  $G_\sigma$  is  $G_\tau$  plus the leaf  $n + 1$  attached to  $\tau_{N-1}$ . Hence  $G_\tau$  is also a tree; that is,  $\tau \in \mathcal{T}_n$ . Every vertex of degree at least 2 in  $G_\tau$  must also be of degree at least 2 in  $G_\sigma$ . Because  $\sigma_j = \tau_j$  for  $1 \leq j \leq N - 1$ , applying the inductive hypothesis to  $\tau$  yields the desired value of  $k$ . The only interior vertex of  $G_\sigma$  which might be a leaf in  $G_\tau$ , and thus not covered by the inductive hypothesis, is  $\tau_{N-1}$ . As the sequence  $\tau_{N-1}, n + 1, \tau_{N-1}$  appears in the center of  $\sigma = \text{Rf}(\tau)$ , we take  $k = N$ .

The argument is nearly identical when  $\sigma = S^N(\text{Rf}(\tau))$ ; replace  $\tau_{N-1}$  by  $\tau_1$  and note that  $\sigma_{N+j} = \tau_j$  for  $1 \leq j \leq N - 1$ .

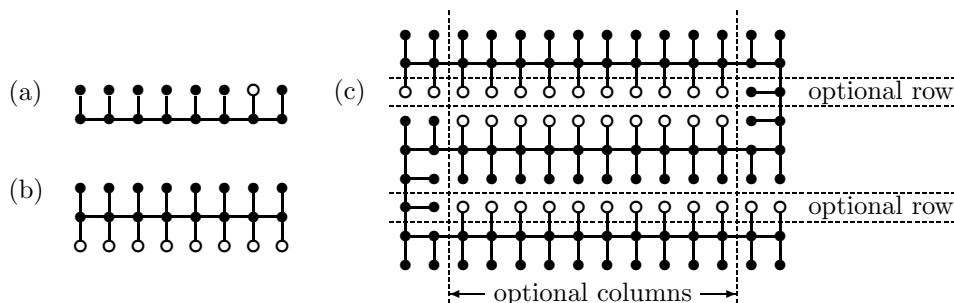


FIGURE 4. Cyclically Gray-codable spanning trees of grids. White leaves are added to interior vertices during construction.

Now consider  $\sigma = S^j(\text{Rf}(\tau))$  for some  $j \not\equiv 0, N \pmod{2N}$ . By Proposition 2.1(2),  $G_\sigma$  is  $G_\tau$  plus a new vertex  $n+1$  adjacent to both  $\tau_1$  and  $\tau_{N-1}$ . The connectedness of  $G_\tau$  and  $\sigma \in \mathcal{T}_{n+1}$  imply  $\tau \in \mathcal{T}_n$ ,  $n+1$  is a leaf in  $G_\sigma$ , and  $\tau_1 = \tau_{N-1}$ . By Proposition 2.1(3),  $\rho = \text{Rf}(\tau) \in \mathcal{T}_{n+1}$  is sufficient, so  $G_\sigma = G_\rho$ . By our work in the  $j = 0$  case above applied to  $\rho$ , there must be an  $l$ ,  $1 < l < 2N$ , such that  $\rho_{l-1} = \rho_{l+1} = v$ . Thus  $\sigma_{l-1-j} = \sigma_{l+1-j} = v$ : whenever  $l-j$  is not  $-1, 0$ , or  $1 \pmod{2N}$ , we can take  $k = l-j \pmod{2N}$ . For  $l-j$  congruent to one of  $-1, 0$ , or  $1 \pmod{2N}$ , we consider several cases.

- When  $1 < l < N-1$  or  $N+1 < l < 2N-1$ ,  $\rho = \text{Rf}(\tau)$  ensures that we have a disjoint  $v, u, v$  sequence in the other half of  $\rho$ . Specifically,  $\rho_{2N-l-1} = \rho_{2N-l+1} = v$ , so we take  $k = 2N-l-j \pmod{2N}$ .
- Because  $\rho_N = \rho_0 = n+1$  is a leaf in  $G_\rho = G_\sigma$ , while  $v$  is an interior vertex, it is impossible that  $l = N-1$  or  $l = N+1$ .
- The only remaining case is  $l = N$  while  $j$  is one of  $N-1, N$ , or  $N+1$ . Then  $v = \tau_{N-1}$ . We already know that  $\tau_1 = \tau_{N-1}$ . Thus,  $\rho_{2N-1} = \rho_1 = v$ , and we can take  $k = N+1, N$ , or  $N-1$ , respectively.

□

**Theorem 2.5.** *For  $n, m > 1$ , there exists a supercomposite strongly  $P_n \times P_m$ -compatible code.*

*Proof.* We may assume that  $m \geq n$ . In each case, Proposition 2.4 ensures that the tree constructed is induced by a sufficient supercomposite Gray code.

When  $n = 2$ ,  $\mathbf{P}_4$  (which is a strongly  $C_4$ -compatible code) covers  $m = 2$ . When  $m \geq 3$ , adding a single leaf to an interior vertex of the graph of  $\mathbf{L}_{m+1}$  yields a *comb*, which spans  $P_2 \times P_m$ . See Figure 4(a). When  $n = 3$  and  $m \geq 3$ , we construct a spanning tree of  $P_3 \times P_m$  by attaching  $m$  leaves to interior vertices of a comb. See Figure 4(b). When  $n, m \geq 4$ , we use a comb as the foundation of a spanning tree. See Figure 4(c). First, zig-zag a comb through the grid. If  $n \equiv 2 \pmod{3}$ , omit one optional row; if  $n \equiv 1 \pmod{3}$ , omit both. Finally, add leaves to interior vertices to fill out the grid. □

**2.3. Trees not induced by supercomposite Gray codes.** Extending the reasoning of our failed effort to find supercomposite Gray codes inducing long paths yields a class of trees that are not induced by any supercomposite Gray code.

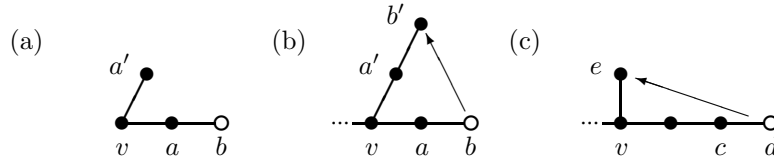


FIGURE 5. Stages in the proof of Theorem 2.6. After white vertices are added, the code is insufficient.

**Theorem 2.6.** *When a tree  $T$  contains an interior vertex all of whose neighbors have degree exactly 2 and which is distance at least 3 from all leaves,  $T$  is induced by no supercomposite Gray code.*

*Proof.* Assume there exists a supercomposite Gray code inducing a tree  $T$  with such a vertex  $v$ . Then  $v$  has degree at least 2. At most one neighbor of  $v$ —and thus at most one branch off  $v$ —precedes  $v$  in the construction. Let  $b$  be the first vertex at distance 2 from  $v$  appearing after  $v$  and let  $a$  be the common neighbor of  $b$  and  $v$ .

When  $b$  is added,  $a$  is a leaf. We cannot be done when  $b$  is added; there would be a branch off  $v$  of length only 2.

- If  $b$ ,  $v$ , and  $a$  are the only vertices present,  $v$  is still a leaf, so some neighbor  $a'$  must be attached to  $v$  as a leaf later in the construction. See Figure 5(a).
- Otherwise at least 3 vertices precede  $b$  in the construction. Because  $b$  is the first vertex attached to a neighbor of  $v$ , case (1) of Proposition 2.2 applies. The code is insufficient after  $b$  is added; by Proposition 2.1(2), we must either reflect, or half-shift, then reflect, to avoid forming a 4-cycle. In either case, the next vertex added,  $b'$ , is a leaf off of a neighbor  $a'$  of  $v$ . See Figure 5(b).

For  $v$  to be distance at least 3 from all leaves in  $T$ , we must add leaves to both  $b$  and  $b'$ . We conclude that *at least 2 vertices at distance 3 from  $v$  appear after  $v$ .*

Let  $d$  be the first vertex at distance 3 from  $v$  appearing after  $v$  and let  $c$  be its neighbor. At least one more vertex at distance 3 from  $v$  must be added. Since  $c$  and  $v$ 's common neighbor has degree 2,  $v$  is the only vertex at distance 2 from  $c$ . By Proposition 2.2, we must precede or follow  $d$  by adding a leaf  $e$  to  $v$ . See Figure 5(c).

In fact, every time we add an initial leaf to a vertex at distance 2 from  $v$ , the assumption that all neighbors of  $v$  have degree 2 forces us to start a new branch off of  $v$  at the preceding, or following, step. We can never catch up and complete all the branches, so no supercomposite  $T$ -compatible code can exist.  $\square$

**Corollary 2.7.** *When a tree on  $n$  vertices has  $(n + 13)/10$  or fewer leaves, it is induced by no supercomposite Gray code.*

*Proof.* First, we argue that *a tree with  $k$  leaves is topologically equivalent to a tree on at most  $2k - 2$  vertices.* Paths are topologically equivalent to 2-vertex trees. When  $k > 2$ , let  $a$  be the average degree of the vertices of degree at least 3 and let  $m$  be the size of a minimal topologically equivalent tree. Then  $2(m - 1) = k + a(m - k)$ , and  $m$  is maximized when  $a \geq 3$  is minimized. Thus  $m \leq 2k - 2$ .

Given an  $n$ -vertex tree  $T$  with  $k \leq (n + 13)/10$  leaves, let  $T'$  be a topologically equivalent tree with at most  $(n + 3)/5$  vertices. In constructing  $T$  from  $T'$ , at least

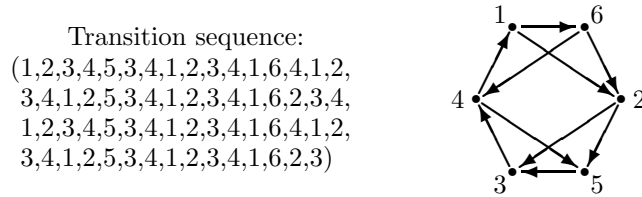


FIGURE 6. A cyclic code whose graph lacks bidirectional edges.

$(4n - 3)/5$  vertices of degree 2 must be inserted into the at most  $(n - 2)/5$  edges of  $T'$ ; thus, at least 5 vertices of degree 2 must be inserted into some edge of  $T'$ , and the condition of our initial claim is satisfied.  $\square$

### 3. DIGRAPHS

The *digraph*  $D_{\mathbf{B}}$  induced by an  $n$ -bit Gray code  $\mathbf{B}$  has vertex set  $[n]$ ; its edges are the ordered pairs  $(\tau_i, \tau_{i+1})$ , where  $\tau(\mathbf{B}) = (\tau_1, \tau_2, \dots, \tau_{N-1})$ . When  $\mathbf{B}$  is cyclic with closing transition  $\tau_N$ ,  $\overline{D}_{\mathbf{B}}$  also contains the directed edges  $(\tau_{N-1}, \tau_N)$  and  $(\tau_N, \tau_1)$ . Bultena and Ruskey [1] introduce digraphs of Gray codes, note that cyclic digraphs of cyclic codes must be strongly connected, and ask whether it is possible for a cyclic code to induce a digraph containing no bidirectional edges.

**Theorem 3.1.** *For every  $n \geq 6$ , there exists a cyclic  $n$ -bit Gray code whose cyclic graph contains no bidirectional edges.*

*Proof.* We use induction on the number of vertices. Figure 6 shows a suitable code on 6 bits (there are 338 such codes; this is the unique edge-minimal one). Let  $\tau = (\tau_1, \tau_2, \dots, \tau_{N-1})$  be the transition sequence of a cyclic  $n$ -bit Gray code whose cyclic graph contains no bidirectional edges. Consider the transition sequence

$$\sigma = (\tau_1, \tau_2, \dots, \tau_{N-1}, n + 1, \tau_1, \tau_2, \dots, \tau_{N-1}).$$

Proposition 1.1 ensures that  $\sigma$  generates a cyclic code with closing transition  $n + 1$ . The only edges of  $\overline{G}_{\sigma}$  not in  $\overline{G}_{\tau}$  are  $(\tau_{N-1}, n + 1)$  and  $(n + 1, \tau_1)$ . Because  $\overline{G}_{\tau}$  includes the edges  $(\tau_{N-1}, \tau_N)$  and  $(\tau_N, \tau_1)$ ,  $\tau_1 \neq \tau_{N-1}$ ; therefore,  $\overline{G}_{\sigma}$  also contains no bidirectional edges.  $\square$

*Remark.* Gilbert [3] observed that the concatenation of the transition sequences of any two  $n$ -bit Gray codes, separated by  $n + 1$ , generates an  $(n + 1)$ -bit code. Reflection and the construction in the last proof are both special cases.

### 4. DISCUSSION AND OPEN QUESTIONS

It is a wide open question to characterize the set of graphs  $G$  for which  $G$ -compatible codes, or strongly  $G$ -compatible cyclic codes, exist. Little is known, and exhaustive searches are prohibitively long for graphs on as few as 8 vertices. Because codes induce connected graphs, trees, as edge-minimal connected graphs, are of particular interest. The standard reflected  $n$ -bit Gray code induces the star  $K_{1, n-1}$  [1, 8]. While  $P_n$ -compatible codes exist for  $n \leq 6$ , computation has verified that no  $P_7$ -compatible code exists, and it is conjectured that no  $P_n$ -compatible code exists for any larger value of  $n$  [1, 8].



FIGURE 7. Trees of diameter 3 with and without a degree 2 vertex.

Some results are known for graphs outside these simple families. Bultena and Ruskey [1] show that every diameter 4 tree is the cyclic graph of some cyclic Gray code and catalogue the existence or non-existence of  $G$ -compatible codes for all graphs  $G$  on 6 or fewer vertices. Bultena and Ruskey also show that when  $T$  is a diameter 3 tree, no strongly  $T$ -compatible cyclic code exists. Theorem 4.1, although developed independently [11], refines Bultena and Ruskey’s result to obtain the first infinite family of connected graphs known not to be induced by Gray codes.

**Theorem 4.1.** *Let  $T$  be a tree of diameter 3. When  $T$  has a degree 2 vertex, there are no strongly  $T$ -compatible cyclic codes. When  $T$  has no degree 2 vertex, there are no  $T$ -compatible codes.*

*Proof.* Because  $T$  has diameter 3,  $n = |V(T)| \geq 4$ . When  $n = 4$ ,  $T$  is  $P_4$  and has two degree 2 vertices. As can be verified computationally (or by checking Gilbert’s [3] list of isomorphism types of 4-bit Gray codes), all  $P_4$ -compatible codes are isomorphic to  $\mathbf{P}_4$ , which is not strongly  $P_4$ -compatible. Thus, for the rest of the proof, we assume  $n \geq 5$ . Let  $\mathbf{B}$  be a  $T$ -compatible code with  $\tau(\mathbf{B}) = (\tau_1, \tau_2, \dots, \tau_{N-1})$ . Label the centers of  $T$  1 and 2. See Figure 7.

Given a codeword  $\mathbf{b} = b_1 b_2 \dots b_n$ , let  $l(\mathbf{b}) = b_3 b_4 \dots b_n$  be the *leaf setting* of  $\mathbf{b}$ ;  $l(\mathbf{b})$  includes bits in positions corresponding to leaves.

- When  $\tau_j$  is 1 or 2, and  $\tau_{j-1}$  and  $\tau_{j+1}$  are both leaves, then  $l(\mathbf{b}_j) = l(\mathbf{b}_{j+1})$ , but  $l(\mathbf{b}_{j-1})$  and  $l(\mathbf{b}_{j+2})$  are both different from  $l(\mathbf{b}_j)$ . That is, exactly two consecutive codewords have the same leaf setting.
- When 1, 2 or 2, 1 is preceded and followed by leaves, exactly three consecutive codewords have the same leaf setting. Call these subsequences *crossings*.
- When 1, 2, 1 or 2, 1, 2 is preceded and followed by leaves, exactly four consecutive codewords have the same leaf setting.
- By Proposition 1.1, there cannot be four or more consecutive transitions at the centers.
- When a leaf starts (respectively, ends) the transition sequence, the code starts (respectively, ends) with a single codeword whose leaf setting differs from that of its successor (respectively, predecessor).

Whenever  $\tau_j$  is a leaf and  $2 \leq j \leq N - 2$ , both  $\tau_{j-1}$  and  $\tau_{j+1}$  are the center which is  $\tau_j$ ’s neighbor. Although  $l(\mathbf{b}_j) \neq l(\mathbf{b}_{j+1})$ , we have  $l(\mathbf{b}_j) = l(\mathbf{b}_{j-1})$  and  $l(\mathbf{b}_{j+1}) = l(\mathbf{b}_{j+2})$ . The *only* way a codeword can fail to have a neighboring codeword of the same leaf setting is when a leaf begins or ends the transition sequence.

There are exactly four codewords in  $\mathbf{B}$  with each leaf setting. Whenever  $\tau(\mathbf{B})$  contains a crossing, a set of three codewords with the same leaf setting must be completed by a single codeword with that setting. Because isolated codewords occur only at the ends of the code, *there are at most two crossings*. Furthermore, *when there are two crossings, the transition sequence must start and end with leaves*.

Assume first that center 2 has degree 2 and is adjacent to leaf 3 (see Figure 7). The leaf 3 must appear at least once in  $\tau$ .

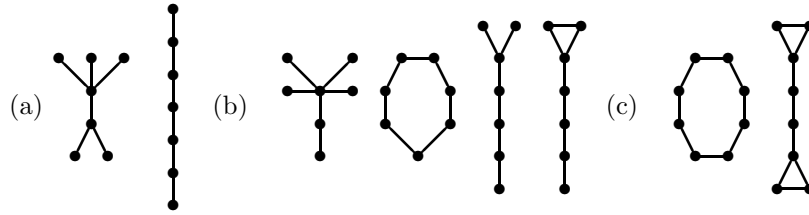


FIGURE 8. (a) 7-vertex trees induced by no Gray code. (b) 7-vertex graphs which are not cyclic graphs of cyclic codes. (c) Maximal 8-vertex graphs not known to be induced by codes.

- If 3 appears only once, the  $N/2$  codewords of each parity in position 3 ensure that  $\tau_{N/2} = 3$ . Because every codeword is used, each leaf adjacent to 1 must be visited both before and after 3. Thus, there must be at least two crossings, and the code must start and end at leaves—necessarily adjacent to 1, as 3 does not appear again. If  $\tau$  is cyclic, Proposition 1.1 implies its closing transition is 3. But then  $\overline{G}_\tau$  contains an edge joining 3 to a leaf adjacent to 1, so  $\tau$  is not strongly  $T$ -compatible.
- Now assume 3 appears at least twice in  $\tau$  and some leaf  $l$  adjacent to 1 occurs between two visits to 3. Because there must be a crossing between 3 and  $l$  and another between  $l$  and 3, at most one 3 – 3 interval contains such a leaf  $l$ . Furthermore, those two crossings imply that the code must start and end with leaves; indeed, with 3, since there can be no additional crossings. Proposition 1.1 restricts  $\tau$  to the following forms (where 3 does not appear in the elided segments):  $3 \dots l \dots 3$ ,  $32123 \dots l \dots 3$ ,  $3 \dots l \dots 32123$ , or  $32123 \dots l \dots 32123$ . Each has 8 or fewer codewords with bit 3 parity different from that of the center section. But  $n \geq 5$  and  $8 < 2^5/2$ , so there is no such  $\tau$ .
- What if 3 occurs at least twice in  $\tau$ , but no leaf adjacent to 1 appears between 3's? By Proposition 1.1, every 3 – 3 segment in  $\tau$  must be 32123. The fact that every vertex appears in  $\tau$ , together with Proposition 1.1, implies that there must be a leaf  $l$  adjacent to 1 such that  $\tau$  starts with 3212321 $l$  or ends with  $l$ 1232123 and  $\tau$  contains no other 3's. But then we have only 4 codewords of one parity in bit 3. As  $n \geq 5$  and  $4 < 2^5/2$ , no such code exists.

When 1 and 2 both have degree at least 3, it is not possible to obtain all possible leaf settings with only two crossings. Thus, no  $T$ -compatible codes exist.  $\square$

Figures 8(a) and 8(b) extend the catalogue of Gray-compatibility begun in [1] to 7-vertex graphs. For every 7-vertex connected graph  $G$  not shown, there exists a strongly  $G$ -compatible cyclic code. The only 8-vertex graphs  $G$  for which it is known that no  $G$ -compatible codes exist are those given by Theorem 4.1. Figure 8(c) shows some maximal 8-vertex graphs  $G$  for which no  $G$ -compatible codes are known. (The computer programs used to produce these results are available from the authors.)

Paths seem to place very stringent restrictions on Gray codes; do cycles offer enough freedom? Bultena and Ruskey [1] ask whether a strongly  $C_n$ -compatible cyclic code exists for any  $n > 5$ . Exhaustive search reveals that there are 54  $C_6$ -compatible codes (up to isomorphism), none of which are strongly  $C_6$ -compatible, while there are only 30  $C_7$ -compatible codes—again, none are strongly  $C_7$ -compatible.

*Question.* Is there an  $n_0$  such that no  $C_n$ -compatible code exists for any  $n > n_0$ ?

Even though every tree on 7 or fewer vertices induced by a Gray code is in fact induced by a supercomposite Gray code, the pattern seems unlikely to hold.

*Question.* Does there exist a tree  $T$  induced by some Gray code, but by no supercomposite Gray code?

The Gray codes with no bidirectional edges constructed in Theorem 3.1 can also be described as inducing directed graphs of directed girth greater than 2.

*Question.* How large can the digirth of the digraph of a Gray code be?

Attention has so far focused on sparse graphs—the fewer edges, the more restrictions. Shouldn't typical Gray codes induce many edges? The largest code we know of which induces a complete graph is an 8-bit code appearing in [10].

*Problem.* Construct  $n$ -bit Gray codes which induce  $K_n$ .

#### ACKNOWLEDGMENTS

In 1989 and 1990, the first author attended the University of Minnesota, Duluth Research Experiences for Undergraduates program, supported by the National Science Foundation (DMS-9000742) and the National Security Agency (MDA 904-88-H-2027). Joseph Gallian, director of the Duluth REU, suggested this problem and provided much encouragement. The second author has been supported by an IBM Cooperative Fellowship. We thank Lenore Cowen, David Moews, and David Witte for their helpful comments on our manuscript. We are grateful to the anonymous referees whose suggestions greatly improved our proofs and presentation.

#### REFERENCES

- [1] B. Bultena and F. Ruskey, Transition restricted Gray codes, *Electron. J. Combin.* 3(1996) #R11.
- [2] J.H. Conway, N.J.A. Sloane, and A.R. Wilks, Gray codes for reflection groups, *Graphs Combin.* 5(1989) 315–325.
- [3] E.N. Gilbert, Gray codes and paths on the  $n$ -cube, *Bell System Tech. J.* 37(1958) 815–826.
- [4] L. Goddyn, G.M. Lawrence, and E. Nemeth, Gray codes with optimized run lengths, *Util. Math.* 34(1988) 179–192.
- [5] F. Gray, Pulse code communication, U. S. Patent 2,632,058, 1958.
- [6] C. Savage, A survey of combinatorial Gray codes, *SIAM Rev.* 39(1997) 605–629.
- [7] P.J. Slater, Open problem, in: *Proc. 10th Southeastern Conference on Combinatorics, Graph Theory, and Computing*, Vol. II (Utilitas Mathematica Publishing, Winnipeg, 1979) 918–919.
- [8] P.J. Slater, Research Problems 109 and 110, *Discrete Math.* 76(1989) 293–294.
- [9] P.J. Slater, personal communication, 1989.
- [10] V.E. Vickers and J. Silverman, A technique for generating specialized Gray codes, *IEEE Trans. Comput.* C-29(1980) 329–331.
- [11] E.L. Wilmer, Gray codings of trees, Undergraduate thesis, Department of Mathematics, Harvard University, 1991.

DEPARTMENT OF MATHEMATICS, OBERLIN COLLEGE, OBERLIN, OH 44074  
*E-mail address:* [elizabeth.wilmer@oberlin.edu](mailto:elizabeth.wilmer@oberlin.edu)

MIT LAB FOR COMPUTER SCIENCE, 77 MASSACHUSETTS AVENUE, CAMBRIDGE, MA 02139  
*E-mail address:* [mernst@lcs.mit.edu](mailto:mernst@lcs.mit.edu) *URL:* <http://sdg.lcs.mit.edu/~mernst/>