Hamilton Cycles in Restricted Rotator Graphs

Brett Stevens[1]* and Aaron Williams[1]

[1] brett@math.carleton.ca Carleton University, CANADA
[2] haron@uvic.ca Carleton University, CANADA

Abstract. The rotator graph has vertices labeled by the permutations of n in one line notation, and there is an arc from u to v if a prefix of u's label can be rotated to obtain v's label. In other words, it is the directed Cayley graph whose generators are $\sigma_k := (1\ 2\ \cdots\ k)$ for $2 \leq k \leq n$ and these rotations are applied to the indices of a permutation. In a restricted rotator graph the allowable rotations are restricted from $k \in \{2, 3, \ldots, n\}$ to $k \in G$ for some smaller (finite) set $G \subseteq \{2, 3, \ldots, n\}$. We construct Hamilton cycles for $G = \{n-1, n\}$ and $G = \{2, 3, n\}$, and provide efficient iterative algorithms for generating them. Our results start with a Hamilton cycle in the rotator graph due to Corbett (IEEE Transactions on Parallel and Distributed Systems 3 (1992) 622–626) and are constructed entirely from two sequence operations we name 'reusing' and 'recycling'.

1 Introduction

Let Π_n denote the set of permutations of $[n] := \{1, 2, \ldots, n\}$ written in one-line notation as strings. For example, $\Pi_3 = \{1\,2\,3,\ 1\,3\,2,\ 2\,1\,3,\ 2\,3\,1,\ 3\,1\,2,\ 3\,2\,1\}$ and we henceforth omit spaces between individual symbols when appropriate. The operation σ_k is a *prefix-rotation*, or simply *rotation*, and it cyclically moves the first k symbols one position to the left. In other words, σ_k applies the permutation $(1\ 2\ \cdots\ k)$ to the indices of a string. For example, $541362\ \sigma_4 = 413562$ since 413 moves one position to the left and 5 "wraps around" into the fourth position. The operation is also known as a *prefix-shift of length* k in the literature.

1.1 Rotator Graphs and Hamilton Cycles

The *rotator graph* $\mathcal{R}_n$ has nodes labeled with the strings in Π_n, and arcs labeled σ_k directed from $\alpha \in \Pi_n$ to $\beta \in \Pi_n$ when $\beta = \alpha\ \sigma_k$. In group-theoretic terms, $\mathcal{R}_n$ is the directed Cayley graph $\overrightarrow{\mathsf{Cay}}(\{\sigma_2, \sigma_3, \ldots, \sigma_n\}, \mathbb{S}_n)$ with generators σ_k for $2 \leq k \leq n$ and where $\mathbb{S}_n$ is the symmetric group corresponding to Π_n. A *restricted rotator graph* for $G \subseteq [n]$ is $\mathcal{R}_n(G) = \overrightarrow{\mathsf{Cay}}(G, \mathbb{S}_n)$ where the generators are restricted to σ_k for $k \in G$. Figure 1 (a) illustrates $\mathcal{R}_3$.

A Hamilton cycle of $\mathcal{R}_n(G)$ can be described by a *Hamilton sequence* of integers $S = s_0, s_1, \cdots, s_{n!-1}$ where σ_{s_i} is the label of the $(i+1)$st arc in the

* Research supported in part by NSERC

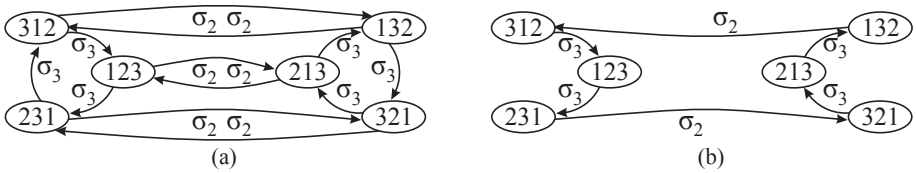

Fig. 1. (a) The rotator graph $\mathcal{R}_3$, and (b) a Hamilton cycle in $\mathcal{R}_3$.

cycle and $s_i \in G$ for each $i \in \{0, 1, \ldots, n! - 1\}$. A Hamilton cycle of $\mathcal{R}_n(G)$ can also be described by the order of node labels along the cycle. In combinatorial generation, these orders are *cyclic Gray codes* since each string in Π_n appears exactly once, and successive strings differ by some σ_k for $k \in G$ where 'successive' includes last to first. For example, Figure 1 (b) contains a Hamilton cycle for $\mathcal{R}_3$ that can be described by

$$3, 3, 2, 3, 3, 2 \quad \text{or} \quad 321, 213, 132, 312, 123, 231. \tag{1}$$

Restricted rotator graphs are vertex-transitive; our Hamilton cycles and their associated Gray code orders for $\mathcal{R}_n(G)$ will all 'start' at $n\ n{-}1\ \cdots\ 1$. Orders of strings that do not necessarily have the Gray code properties are called *lists*.

An explicit Hamilton cycle in $\mathcal{R}_n$ was first constructed by Corbett [2]. Hamilton cycles were then constructed for different generalizations of $\mathcal{R}_n$ by Ponnuswamy and Chaudhary [11] and Williams [13]. Hamilton cycle constructions for $\mathcal{R}_n(\{n-1, n\})$ were proposed as an open problem by Knuth, and this was answered by Ruskey and Williams [12]. Observe that σ_n must be included in a restricted rotator graph in order to generate the entire symmetric group $\mathbb{S}_n$. Moreover, σ_n and σ_c are not sufficient for generating $\mathbb{S}_n$ when c is an arbitrary fixed value. For example, the parity of a permutation cannot be changed if n and c are both odd. A well-known conjecture is that a Hamilton cycle exists in every connected undirected Cayley graph, where undirected Cayley graphs include the inverse of each generator. In particular, a Hamilton cycle was constructed for $\mathsf{Cay}(\{\sigma_2, \sigma_n\}, \mathbb{S}_n)$ by Compton and Williamson in a 50-page paper [1].

Corbett introduced the term "rotator graph" when considering point-to-point multiprocessor networks, where Hamilton cycles establish indexing schemes for sorting and for mapping rings and linear arrays [2] Applications of rotator graphs include fault-tolerant file transmission by Hamada et al [5] and parallel sorting by Corbett and Scherson [3]. Properties of rotator graphs have been examined including minimum feedback sets by Kuo et al [9] and node-disjoint paths by Yasuto, Ken'Ichi, and Mario [14]. Other variations of rotator graphs include incomplete rotator graphs [11], the bi-rotator graph (see Lin and Hsu [10]), and graphs where the labels can have repeated symbols [13]. The relationship between Hamilton cycles of $\mathcal{R}_n(n-1, n)$ and universal cycles of Π_n is discussed by Holroyd, Ruskey, and Williams along with applications [6, 7].

1.2 New Results

We construct a new Hamilton cycle in $\mathcal{R}_n(\{n-1, n\})$ and the first Hamilton cycle in $\mathcal{R}_n(\{2, 3, n\})$. The new constructions are intimately related to Corbett's original Hamilton cycle in $\mathcal{R}_n$. In fact, the beauty of our results is that all three Hamilton sequences can be described by two operations that we name 'reusing' and 'recycling'. We also provide an algorithm for constructing the Hamilton sequences. The algorithm is *loopless* since successive values in the sequence are obtained in worst-case O(1)-time (see Ehrlich [4] for the first use of this term).

Section 2 formally defines the 'reuse' and 'recycle' operations. Section 3 constructs the three Hamilton cycles and proves that two of the constructions are correct. Section 4 gives a loopless algorithm that generates the Hamilton sequences. Section 5 extends Corbett's recursive construction with an iterative description that is instrumental to the final proof of correctness. Section 6 completes the final proof of correctness by proving that Corbett's Hamilton sequence of $\mathcal{R}_n$ can be 'recycled' into a Hamilton cycle of $\mathcal{R}_{n+1}(\{n, n+1\})$. Section 7 concludes with open problems.

2 Sequence Building

This section defines two operations for building sequences of positive integers and examines the lists they create when they are treated as rotation indices.

2.1 Reusing and Recycling

In this subsection we define the reusing and recycling sequence operations, and describe how they are applied to create lists of strings. Given i and n satisfying $1 < i < n$, the result of *reusing* and *recycling* i with respect to n is

$$\mathsf{reuse}_n(i) = \overbrace{n, \ldots, n}^{n-1 \text{ copies}}, n-i+1 \text{ and } \mathsf{recycle}_n(i) = n, n, \overbrace{n-1, \ldots, n-1}^{i-1 \text{ copies}}, \overbrace{n, \ldots, n}^{n-i-1 \text{ copies}}$$

respectively. Notice that both operations create sequences of n symbols that are each at least 2 and at most n. For example,

$$\mathsf{reuse}_6(3) = 6, 6, 6, 6, 6, 4 \text{ and } \mathsf{recycle}_6(3) = 6, 6, 5, 5, 6, 6. \tag{2}$$

We build longer sequences by applying these operations to each symbol in a sequence. If $S = s_1, s_2, \ldots, s_t$ is a sequence with $1 < s_i < n$ for each i, then

$$\begin{aligned} \mathsf{reuse}_n(S) &= \mathsf{reuse}_n(s_1), \mathsf{reuse}_n(s_2), \ldots, \mathsf{reuse}_n(s_t) \text{ and} \\ \mathsf{recycle}_n(S) &= \mathsf{recycle}_n(s_1), \mathsf{recycle}_n(s_2), \ldots, \mathsf{recycle}_n(s_t). \end{aligned}$$

We use sequences to create lists of strings by applying successive prefix-rotations. If $\alpha \in \Pi_n$ and $S = s_1, s_2, \ldots, s_t$ is a sequence with $1 < s_i \leq n$ for each i, then

$$\alpha \circ S = \beta_0, \beta_1, \ldots, \beta_t \text{ where } \beta_0 = \alpha \text{ and } \beta_i = \beta_{i-1}\ \sigma_{s_i} \text{ for } i = 1, 2, \ldots, t.$$

For example, if $\alpha = 612345$ then

$$\begin{aligned} \alpha \circ \mathsf{reuse}_6(3) &= 612345, 123456, 234561, 345612, 456123, 561234, 612534 \quad (3) \\ \alpha \circ \mathsf{recycle}_6(3) &= 612345, 123456, 234561, 345621, 456231, 562314, 623145 \end{aligned}$$

since $\mathsf{reuse}_6(3) = 6, 6, 6, 6, 6, 4$ and $\mathsf{recycle}_6(3) = 6, 6, 5, 5, 6, 6$ by (2). In some situations it is more convenient to leave off the last permutation in the list $\alpha \circ S$, and we use $\alpha \bullet S$ in these cases.

A symbol x is *periodic* in a list L of Π_n if the position of x moves once to the left (cyclically) between successive strings in L. For example, 6 is periodic in both lists from (3). More generally, the first symbol x of $\alpha \in \Pi_n$ is periodic in any list of the form $\alpha \circ \mathsf{reuse}(S)$ or $\alpha \circ \mathsf{recycle}(S)$. This is because the first rotation σ_n moves x from the first position to the last position, the next $n-1$ rotations move x one position to the left, and this pattern is repeated.

2.2 Rotation Identities

In this subsection we give two identities involving rotations. In addition to $\sigma_i = (1\ 2\ \cdots\ i)$ for prefix-rotations, let $\varsigma_i = (n{-}i{+}1\ n{-}i{+}2\ \cdots\ n)$ denote the *suffix-rotation* operation, and $\sigma'_i = (2\ 3\ \cdots\ i{+}1)$ denote a modified prefix-rotation that begins at the second symbol. We also let σ_i^j denote j successive copies of σ_i, and successive rotations are applied from left-to-right. Using these conventions we have the following simple identities

$$\underset{\text{“reuse equality”}}{\sigma_n^{n-1}\sigma_{n-i+1} = \varsigma_i} \quad \text{and} \quad \underset{\text{“recycle equality”.}}{\sigma_n^2\sigma_{n-1}^{i-1}\sigma_n^{n-i-1} = \sigma'_i} \qquad (4)$$

The “reusing equality” on the left follows from $(n\ n{-}1\ \cdots\ 1)(1\ 2\ \cdots\ n{-}i{+}1) = (n{-}i{+}1\ n{-}i{+}2\ \cdots\ n)$, while the “recycling equality” on the right is the second equality of Lemma 2 in [7]. The equalities allow the last string obtained by applying $\mathsf{reuse}_n(i)$ and $\mathsf{recycle}_n(i)$ to be computed directly. For example, when $i = 3$ and $n = 6$ we obtain the final strings in (3) as follows

$$\begin{aligned} 612345\sigma_6^5\sigma_4 &= 612345\varsigma_3 & 612345\sigma_6^2\sigma_5^2\sigma_6^2 &= 612345\sigma'_3 \quad (5) \\ &= 612534 & &= 623145. \end{aligned}$$

2.3 List Quotients

In Section 2.1 we saw that every nth string in $n\ n{-}1\ \cdots\ 1 \circ S$ begins with n, whenever S is obtained by reusing or recycling. Furthermore, Section 2.2 gave identities for these strings. This subsection examines these strings in more detail.

The *quotient* of a list L of Π_n with a symbol $x \in [n]$ is the list obtained from L by (1) removing the strings that do not begin with x, and (2) removing x from the strings that begin with x. We denote this operation by x/L. Our first lemma uses recycling and is illustrated by the next example. If $S = 3, 3, 2, 3, 3, 2$ then

$$\begin{aligned} 321 \bullet S &= \underline{321}, 213, 132, 312, 123, 231 \text{ and} \quad (6) \\ 4321 \bullet \mathsf{recycle}(S) &= 4\underline{321}, 3214, 2143, 1423, 4\underline{213}, 2134, 1342, 3412, 4\underline{132}, 1324, 3241, 2431, \\ &\quad 4\underline{312}, 3124, 1243, 2413, 4\underline{123}, 1234, 2341, 3421, 4\underline{231}, 2314, 3142, 1432. \end{aligned}$$

Notice the quotient of the second list with 4 equals the first list (as underlined). That is, $4/(4321 \bullet \mathsf{recycle}(S)) = 321 \bullet S$. Lemma 1 proves this is true for any S.

Lemma 1. *If sequence S has values in $\{2, \ldots, n{-}1\}$ and $\alpha_i = i\ i{-}1\ \cdots\ 1$, then*

$$n/(\alpha_n \bullet \mathsf{recycle}_n(S)) = \alpha_{n-1} \bullet S.$$

Proof. The first string in both lists is α_{n-1} since $n/\alpha_n = \alpha_{n-1}$. Since n is periodic in $\alpha_n \circ \mathsf{recycle}_n(S)$, every nth string begins with n. Therefore, successive strings in $n/(\alpha_n \circ \mathsf{recycle}_n(S))$ are obtained by successive σ_{s_i} for $S = s_1, \ldots, s_t$ by the "recycling identity" in (4). Therefore, the two lists are equal. □

Our second lemma instead uses reusing and is illustrated by the next example

$$\begin{aligned} 321 \bullet S &= \underline{321}, \underline{213}, \underline{132}, \underline{312}, \underline{123}, \underline{231} \text{ and} \qquad (7)\\ 4321 \bullet \mathsf{reuse}(S) &= 4\underline{321}, 3214, 2143, 1432, 4\underline{132}, 1324, 3241, 2413, 4\underline{213}, 2134, 1342, 3421,\\ &\quad\ 4\underline{231}, 2314, 3142, 1423, 4\underline{123}, 1234, 2341, 3412, 4\underline{312}, 3124, 1243, 2431. \end{aligned}$$

In this case the quotient of the second list with 4 equals the "double-reverse" of the first list. Given a string $a_1 a_2 \cdots a_n \in \mathit{\Pi}_n$ the *double-reverse* is

$$a_1 a_2 \cdots a_n^R = (n{-}a_n{+}1)\ \cdots\ (n{-}a_2{+}1)\ (n{-}a_1{+}1).$$

In a double-reverse the relative order of symbols is changed from $a_1 a_2 \cdots a_n$ to $a_n \cdots a_2 a_1$ and relative values are reversed from x to $n{-}x{+}1$. Given a list $L = \alpha_1, \ldots, \alpha_m$ the *double-reversal* of L is $L^R = \alpha_1^R,\ \ldots,\ \alpha_m^R$. For example,

$$\begin{aligned} (321, 132, 213, 231, 123, 312)^R &= 321^R, 132^R, 213^R, 231^R, 123^R, 321^R\\ &= 321, 213, 132, 312, 123, 231. \end{aligned}$$

This equation illustrates the relationship $4/(4321 \bullet \mathsf{reuse}(S)) = (321 \bullet S)^R$ in (7) (as underlined). Lemma 2 proves this is true for any S.

Lemma 2. *If sequence S has values in $\{2, \ldots, n{-}1\}$ and $\alpha_i = i\ i{-}1\ \cdots\ 1$, then*

$$n/(\alpha_n \circ \mathsf{reuse}_n(S)) = (\alpha_{n-1} \circ S)^R.$$

Proof. The first string in both lists is α_{n-1} since $n/\alpha_n = \alpha_{n-1}$ and $\alpha_{n-1}^R = \alpha_{n-1}$. Since n is periodic in $\alpha_n \circ \mathsf{reuse}_n(S)$, every nth string begins with n. Therefore, successive strings in $n/(\alpha_n \circ \mathsf{reuse}_n(S))$ are obtained by successive ς_{s_i} for $S = s_1, \ldots, s_t$ by the "reusing identity" in (4). Notice that suffix-rotations in a double-reversed string are 'equivalent' to prefix-rotations in the original string. That is, if $\alpha = \beta^R$, then $\alpha\ \sigma_i = \beta\ \varsigma_i^R$. Therefore, the two lists are equal. □

3 Three Hamilton Sequences

This section constructs Hamilton sequences for $\mathcal{R}_n$, $\mathcal{R}_n(\{n{-}1, n\})$, and $\mathcal{R}_n(\{2, 3, n\})$ through reusing and recycling. Two of the three main theorems are proven in this section, and the third is proven in Sections 5 and 6.

3.1 Hamilton Sequence for $\mathcal{R}_n$

This subsection proves that a Hamilton sequence for $\mathcal{R}_n$ can be obtained entirely with the reuse operation. The *Corbett sequence* is defined recursively as follows

$$C(n) = \begin{cases} 1 & \text{if } n = 1 \\ \mathsf{reuse}_n(C(n{-}1)) & \text{if } n > 1. \end{cases} \tag{8}$$

Corbett proved that $C(n)$ is a Hamilton sequence for the rotator graph $\mathcal{R}_n$ [2]. Let $\Pi_C(n) = n\ n{-}1\ \cdots\ 1 \circ C(n)$ denote this *Corbett Gray code* of Π_n. Table 1 gives $C(n)$ and $\Pi_C(n)$ for $n = 3, 4$.

$C(3)$	$C(4) = \mathsf{reuse}_4(C(3))$	$\Pi_C(3)$	$\Pi_C(4) = 4321 \circ C(4)$
3,	4, 4, 4, 2,	$\underline{321}$,	$4\underline{321}$, 3214, 2143, 1432,
3,	4, 4, 4, 2,	$\underline{213}$,	$4\underline{132}$, 1324, 3241, 2413,
2,	4, 4, 4, 3,	$\underline{132}$,	$42\underline{13}$, 2134, 1342, 3421,
3,	4, 4, 4, 2,	$\underline{312}$,	$42\underline{31}$, 2314, 3142, 1423,
3,	4, 4, 4, 2,	$\underline{123}$,	$4\underline{123}$, 1234, 2341, 3412,
2	4, 4, 4, 3	$2\underline{31}$	$43\underline{12}$ 3124, 1243, 2431
(a)	(b)	(c)	(d)

Table 1. (a)-(b) Corbett sequence for $n = 3, 4$, and (c)-(d) Corbett Gray code for $n = 3, 4$. Prefix-rotations in (c) and suffix-rotations of every fourth string in (d) are underlined according to (a) by the "reusing equality" in (4).

Theorem 1 extends Corbett's result by proving that any Hamilton sequence for $\mathcal{R}_{n-1}$ can be 'reused' into a Hamilton sequence for $\mathcal{R}_n$. Furthermore, we explicitly state the values used in the resulting sequence. (A simple induction proves that Corbett's 'canonical' sequence $C(n)$ uses each value in $\{2, 3, \ldots, n\}$.)

Theorem 1. *[2] If S is a Hamilton sequence in $\mathcal{R}_{n-1}(G)$, then $\mathsf{reuse}_n(S)$ is a Hamilton sequence in $\mathcal{R}_n(H)$, where $i \in H$ if and only if $i = n$ or $n - i + 1 \in G$.*

Proof. Let $\alpha = n\ n{-}1\ \cdots\ 1$. By Lemma 2 the nth strings in $\alpha \circ \mathsf{reuse}_n(S)$ form a Gray code for the strings in Π_n that begin with n. Each of these strings is followed by $n{-}1$ applications of σ_n by the definition of $\mathsf{reuse}_n(i)$. Therefore, $\alpha \circ \mathsf{reuse}_n(S)$ contains every string in Π_n and so $\mathsf{reuse}_n(S)$ is a Hamilton sequence. Finally, the values in H follow immediately from the definition of reusing. □

3.2 Hamilton Sequence for $\mathcal{R}_n(\{n{-}1, n\})$

This subsection states that a Hamilton sequence for $\mathcal{R}_n(n{-}1, n)$ can be obtained by recycling Corbett's Hamilton sequence. In other words, a Hamilton sequence for $\mathcal{R}_n(n{-}1, n)$ can be obtained by repeated reusing following by a single recycle. Let $D(n) = \mathsf{recycle}_n(C(n{-}1))$ denote this sequence and $\Pi_D(n) = n\ n{-}1\ \cdots\ 1 \circ D(n)$ denote its Gray code. Table 2 gives $D(n)$ and $\Pi_D(n)$ for $n = 4$.

$C(3)$	$D(4) = \mathsf{recycle}_4(C(3))$	$\Pi_C(3)$	$\Pi_D(4) = 4321 \circ D(4)$
3,	4, 4, 3, 3,	$\underline{321}$,	$4\underline{321}$, 3214, 2143, 1423,
3,	4, 4, 3, 3,	$\underline{213}$,	$4\underline{213}$, 2134, 1342, 3412,
2,	4, 4, 3, 4,	$\underline{132}$,	$4\underline{132}$, 1324, 3241, 2431,
3,	4, 4, 3, 3,	$\underline{312}$,	$4\underline{312}$, 3124, 1243, 2413,
3,	4, 4, 3, 3,	$\underline{123}$,	$4\underline{123}$, 1234, 2341, 3421,
2	4, 4, 3, 4	$\underline{23}1$	$4\underline{23}1$ 2314, 3142, 1432
(a)	(b)	(c)	(d)

Table 2. (a)-(b) Recycling the Corbett sequence and (c)-(d) the Corbett Gray code from $n = 3$ to $n = 4$. Prefix-rotations in (c) and modified prefix-rotations of every fourth string in (d) are underlined according to (a) by the "recycling equality" in (4).

Theorem 2. *If $S = C(n-1)$ is the Corbett sequence for $\mathcal{R}_{n-1}$, then $\mathsf{recycle}_n(S)$ is a Hamilton sequence in $\mathcal{R}_n(\{n-1, n\})$.*

To illustrate the difficulty of Theorem 2, we point out that arbitrary Hamilton sequences for $\mathcal{R}_{n-1}$ cannot be recycled into Hamilton sequences for $\mathcal{R}_n$. For example, consider the following Hamilton sequence for $\mathcal{R}_4$ and its associated Gray code for Π_4

$$\begin{aligned} S &= 4,3,3,2,3,4,2,3,4,2,3,3,4,4,2,3,3,2,3,4,4,4,3,4 \qquad (9)\\ 4321 \circ S &= 4321, 3214, 2134, \underline{1324}, \underline{3124}, 1234, 2341, 3241, 2431, 4312, 3412, 4132,\\ &\quad 1342, 3421, 4213, 2413, 4123, 1243, 2143, 1423, 4231, \underline{2314}, \underline{3142}, 1432. \end{aligned}$$

Observe that 1324 is followed by 1324 $\sigma_2 = 3124$, and that 2314 is followed by 2314 $\sigma_4 = 3142$ in $4321 \circ S$. Therefore, Lemma 1 implies that 51324 is followed by $51324 \bullet \mathsf{recycle}_5(2)$, and 52314 followed by $52314 \bullet \mathsf{recycle}_5(4)$ in the recycled list $54321 \bullet \mathsf{recycle}_5(S)$. These two sublists appear below

$$\begin{aligned} &51324 \bullet \mathsf{recycle}_5(2) && 52314 \bullet \mathsf{recycle}_5(4) \qquad (10)\\ &= 51324 \bullet 5,5,4,5,5 && = 52314 \bullet 5,5,4,4,4\\ &= 51324, 13245, 32451, 24531, \underline{45312} && = 52314, 23145, 31452, 14532, \underline{45312}. \end{aligned}$$

Since both sublists contain 45312, the list $54321 \bullet \mathsf{recycle}_5(S)$ is not a Gray code. Furthermore, the reader can verify that $\mathsf{recycle}_6(\mathsf{reuse}_5(S))$ is also not a Hamilton sequence. In other words, an arbitrary Hamilton sequence S cannot be recycled into a Hamilton sequence, even when S is the result of reusing a previous Hamilton sequence. We prove Theorem 2 by developing results in Sections 5-6.

3.3 Hamilton Sequence for $\mathcal{R}_n(\{2, 3, n\})$

This subsection proves that a Hamilton sequence for $\mathcal{R}_n(\{2, 3, n\})$ can be obtained by recycling and then reusing Corbett's Hamilton sequence. In other words, a Hamilton sequence for $\mathcal{R}_n(2, 3, n)$ can be obtained by repeated reusing followed by a single recycle and then a single reuse. Let $E(n) = \mathsf{reuse}_n(D(n-1))$

denote this sequence and $\Pi_E(n) = n\ n{-}1\ \cdots\ 1 \circ E(n)$ denote its Gray code. More generally, Theorem 3 proves that a Hamilton sequence for $\mathcal{R}_n(\{2,3,n\})$ can be obtained by reusing any Hamilton sequence for $\mathcal{R}_{n-1}(\{n{-}2,n{-}1\})$.

Theorem 3. *If S is a Hamilton sequence in $\mathcal{R}_{n-1}(\{n{-}2,n{-}1\})$, then $\mathsf{reuse}_n(S)$ is a Hamilton sequence in $\mathcal{R}_n(\{2,3,n\})$.*

Proof. By the statement of the theorem, S is a Hamilton sequence for $\mathcal{R}_{n-1}(G)$ for $G = \{n{-}2,n{-}1\}$. By theorem 1, $\mathsf{reuse}_n(S)$ is a Hamilton sequence in $\mathcal{R}_n(H)$ where $H = \{n{-}(n{-}2){+}1, n{-}(n{-}1){+}1, n\} = \{2,3,n\}$. □

4 Loopless Algorithm

In this section we show how to generate each symbol of Corbett's Hamilton sequence $C(n)$ for the rotator graph $\mathcal{R}_n$ in worst-case O(1)-time. Furthermore, our $\mathsf{CorbettLoopless}(n)$ algorithm is significant because

1. It adapts a well-known algorithm for generating multi-radix numbers, and
2. A modification generates Hamilton sequences in $\mathcal{R}_n(\{n{-}1,n\})$ or $\mathcal{R}_n(\{2,3,n\})$.

4.1 Staircase Sequence

The *staircase sequence* $S(n)$ is obtained from repeated applications of the *step* sequence operation as defined below

$$\mathsf{step}_n(i) = \overbrace{n,\ldots,n}^{n-1 \text{ copies}}, i \quad \text{and} \quad S(n) = \begin{cases} 2 & \text{if } n = 1 \\ \mathsf{step}_n(S(n{-}1)) & \text{if } n > 1. \end{cases} \tag{11}$$

The step operation is identical to the reuse operation except the final symbol i has replaced $n{-}i{+}1$. The simplified sequence allows modular conditions such as

$$i \equiv 0 \bmod n, \text{ and } i \equiv 0 \bmod n{-}1, \text{ and } \ldots, \text{ and } i \equiv 0 \bmod j{+}1, \text{ and } i \not\equiv 0 \bmod j$$

to be referenced succinctly. This is illustrated by Lemma 3, which specifies each value of Corbett's sequence in terms of the staircase sequence.

Lemma 3. *If the staircase sequence is $S(n) = s_1, s_2, \ldots, s_{n!}$ and the Corbett sequence is $C(n) = c_1, c_2, \ldots, c_{n!}$ and $n \geq 2$, then for each i satisfying $1 \leq i \leq n!$*

$$s_i = \begin{cases} n & \text{if } i \not\equiv 0 \bmod n \\ n{-}1 & \text{if } i \equiv 0 \bmod n \text{ and } i \not\equiv 0 \bmod n{-}1 \\ n{-}2 & \text{if } i \equiv 0 \bmod n \text{ and } i \equiv 0 \bmod n{-}1 \\ & \text{and } i \not\equiv 0 \bmod n{-}2 \\ \ldots & \ldots \\ 2 & \text{if } i = \frac{n!}{2} \\ 2 & \text{if } i = n!. \end{cases} \quad \text{and } c_i = \begin{cases} n & \text{if } s_i = n \\ 2 & \text{if } s_i = n{-}1 \\ n{-}1 & \text{if } s_i = n{-}2 \\ 3 & \text{if } s_i = n{-}3 \\ n{-}2 & \text{if } s_i = n{-}4 \\ \ldots & \ldots \\ \lceil \frac{n+1}{2} \rceil & \text{if } s_i = 2. \end{cases}$$

Proof. The result is true for $n = 2$ since $S(2) = C(2) = 2, 2$. Assume the result is true for $S(k) = s'_1, \ldots, s'_{k!}$ and $C(k) = c'_0, \ldots, c'_{k!}$ for $k \geq 2$. When $n = k{+}1$,

$$S(n) = \mathsf{step}(S(n{-}1)) = \overbrace{n, \ldots, n}^{n-1 \text{ copies}}, s'_1, \ \ldots, \ \overbrace{n, \ldots, n}^{n-1 \text{ copies}}, s'_{n-1!}$$
$$C(n) = \mathsf{reuse}(C(n{-}1)) = \underbrace{n, \ldots, n}_{n-1 \text{ copies}}, n{-}c'_1{+}1, \ \ldots, \ \underbrace{n, \ldots, n}_{n-1 \text{ copies}}, n{-}c'_{n-1!}{+}1$$

so the result follows by induction by (11) and (8), respectively. □

Algorithm 1 Generate the staircase sequence $S(n)$ by StaircaseLoopless(n) and Corbett's Hamilton sequence $C(n)$ for rotator graph $\mathcal{R}_n$ by CorbettLoopless(n). Note: The final symbol output by StaircaseLoopless(n) is 1 instead of 2 by (11).

Require: StaircaseLoopless(n)
1:
2: $a_1 \cdots a_n \leftarrow 0 \cdots 0$
3: $d_1 \cdots d_n \leftarrow 1 \cdots 1$
4: $f_1 \cdots f_n \leftarrow 1 \cdots n$
5: **loop**
6: $j \leftarrow f_1$
7: output($n{-}j{+}1$)
8: **if** $j = n$ **then**
9: **return**
10: **end if**
11: $f_1 \leftarrow 1$
12: $a_j \leftarrow a_j + d_j$
13: **if** $a_j = 0$ **or** $a_j = n{-}j$ **then**
14: $d_j \leftarrow -d_j$
15: $f_j \leftarrow f_{j+1}$
16: $f_{j+1} \leftarrow j + 1$
17: **end if**
18: **end loop**

Require: CorbettLoopless(n)
1: $r_1 \cdots r_n \leftarrow n\ 2\ n{-}1\ 3 \cdots \lceil \frac{n+1}{2} \rceil\ \lceil \frac{n+1}{2} \rceil$
2: $a_1 \cdots a_n \leftarrow 0 \cdots 0$
3: $d_1 \cdots d_n \leftarrow 1 \cdots 1$
4: $f_1 \cdots f_n \leftarrow 1 \cdots n$
5: **loop**
6: $j \leftarrow f_1$
7: output(r_j)
8: **if** $j = n$ **then**
9: **return**
10: **end if**
11: $f_1 \leftarrow 1$
12: $a_j \leftarrow a_j + d_j$
13: **if** $a_j = 0$ **or** $a_j = n{-}j$ **then**
14: $d_j \leftarrow -d_j$
15: $f_j \leftarrow f_{j+1}$
16: $f_{j+1} \leftarrow j + 1$
17: **end if**
18: **end loop**

4.2 Staircase Strings

Staircase sequences arise naturally in combinatorial generation. A string $\alpha = a_1 a_2 \cdots a_n$ is a *staircase string* if its symbols satisfy $1 \leq a_i \leq i$ for all $1 \leq i \leq n$. In other words, staircase strings are multi-radix numbers with radices $m_i = i$ for $1 \leq i \leq n$. Loopless Algorithm H in *The Art of Computer Programming* generates multi-radix numbers in reflected Gray code order, meaning that successive strings differ by ± 1 in a single symbol (see Knuth [8] pg. 20). In the special case of staircase strings, Algorithm H generates the $\pm$ indices according to the staircase sequence. For example, the Gray code appears below for $n = 3$

$$11\underline{1}, 11\underline{2}, 1\underline{1}3, 12\underline{3}, 12\underline{2}, 1\underline{2}1,$$

where the $\pm$ indices follow $S(3) = 3, 3, 2, 3, 3, 2$ (cyclically). StaircaseLoopless(n) in Algorithm 1 gives our presentation of Algorithm H, which is simplified by removing references to the multi-radix number and by "hard-coding" the radices $m_i = i$ for $1 \le i \le n$. Array d and f store $\pm$ directions and focus pointers, respectively. To generate $C(n)$, we introduce an auxiliary array of constants

$$r_1, r_2, r_3, r_4, \cdots, r_{n-1}, r_n = n, 2, n{-}1, 3, \cdots, \left\lceil \frac{n}{2} \right\rceil, \left\lceil \frac{n}{2} \right\rceil$$

whose values are explained by Lemma 3. Finally, CorbettLoopless(n) in Algorithm 1 is obtained by replacing output$(n{-}j{+}1)$ on line 7 by output(r_j).

Theorem 4. CorbettLoopless(n) *is a loopless algorithm that generates Corbett's sequence* $C(n)$.

By Theorem 2 algorithm CorbettLoopless(n) can instead generate the Hamilton sequence $D(n)$ for $\mathcal{R}_{n+1}(\{n, n{+}1\})$ via recycling by replacing line 7 with

$$\mathsf{output}(n{+}1, n{+}1, \underbrace{n, \ldots, n}_{r_j - 1 \text{ copies}}, \underbrace{n{+}1, \ldots, n{+}1}_{n - r_j \text{ copies}}).$$

Similarly, by Theorem 1 the algorithm can generate the Hamilton sequence $E(n)$ for $\mathcal{R}_{n+2}(\{2, 3, n{+}2\})$ via recycling and reusing by replacing line 7 with

$$\mathsf{output}(\overbrace{n{+}2, \ldots, n{+}2}^{n+1 \text{ copies}}, 2, \overbrace{n{+}2, \ldots, n{+}2}^{n+1 \text{ copies}}, 2, \underbrace{\overbrace{n{+}2, \ldots, n{+}2}^{n+1 \text{ copies}}, 3, \ldots, \overbrace{n{+}2, \ldots, n{+}2}^{n+1 \text{ copies}}, 3}_{r_j - 1 \text{ copies}}, \underbrace{\overbrace{n{+}2, \ldots, n{+}2}^{n+1 \text{ copies}}, 2, \ldots, \overbrace{n{+}2, \ldots, n{+}2}^{n+1 \text{ copies}}, 2}_{n - r_j \text{ copies}}).$$

5 Corbett's Successor Rule

In Section 4 we showed how to generate Corbett's sequence $C(n)$ one symbol at a time, with Algorithm CorbettLoopless(n) creating the entire sequence and requiring two auxiliary arrays. Theorem 5 gives a *successor rule* that describes how each string in Corbett's Gray code $\Pi_C(n)$ can be computed from the previous string without additional state. The theorem is illustrated after its proof.

Theorem 5. *Suppose* $\alpha = a_1 a_2 \cdots a_n \in \Pi_n$. *Let* x *and* y *be the lengths of the longest prefix of the form* $n\ n{-}1\ n{-}2\ \cdots$ *and the longest suffix of the form* $\cdots\ 3\ 2\ 1$ *in* $a_2 a_3 \cdots a_n$, *respectively. The string that follows* α *in* $\Pi_C(n)$ *is*

$$\beta = \begin{cases} \sigma_{y+2}(\alpha) & \text{if } x > y \quad (12a) \\ \sigma_{n-x}(\alpha) & \text{otherwise } (x \le y). \quad (12b) \end{cases}$$

Proof. Suppose Corbett's sequence is $C(n) = c_1, c_2, \ldots, c_{n!}$, Corbett's Gray code is $\Pi_C(n) = \alpha_1, \alpha_2, \ldots, \alpha_{n!}$, and $n \ge 2$. Consider an arbitrary $\alpha_i = a_1 a_2 \cdots a_n$ in

the Gray code. By using Lemma 2 and 3 the following conditions can be proven by induction on n

$$\begin{aligned}
a_2 = n &\iff c_i \neq n, \text{and} \\
a_n = 1 \text{ and } a_2 = n &\iff c_i \notin \{n, 2\}, \text{and} \\
a_3 = n{-}1 \text{ and } a_n = 1 \text{ and } a_2 = n &\iff c_i \notin \{n, 2, n{-}1\}, \text{and} \\
a_{n-1} = 2 \text{ and } a_3 = n{-}1 \text{ and } a_n = 1 \text{ and } a_2 = n &\iff c_i \notin \{n, 2, n{-}1, 3\}, \text{and} \\
\cdots &\iff \cdots \\
\alpha = a_1\ n\ n{-}1\ \cdots\ p{+}2\ p{+}1\ a_{q+1}\ p{-}2\ p{-}3\ \cdots\ 2\ 1 &\iff c_i = p.
\end{aligned}$$

where $p = \lceil \frac{n+1}{2} \rceil$ and $q = \lfloor \frac{n+1}{2} \rfloor$. The rule follows from these conditions. □

For example, if $\alpha = 4\underline{87}5623\overline{1}$ then $x = 2$ and $y = 1$ due to the underlined prefix and overlined suffix of 8756231, respectively. Therefore, the string after α in $\Pi_C(8)$ is $\alpha\ \sigma_3 = 48756231\ \sigma_3 = 87456231$ by (12a) since $x > y$ and $y = 2$.

Theorem 5 also allows the lookup table of size $n!$ to be avoided in Corbett's original application involving point-to-point multiprocessor networks [2].

6 Recycling Corbett's Sequence

In this section we prove a restatement of Theorem 2: *If $S = C(n)$ is the Corbett sequence for $\mathcal{R}_n$, then $\mathsf{recycle}_{n+1}(S)$ is a Hamilton sequence in $\mathcal{R}_{n+1}(\{n, n{+}1\})$.*

Proof. We prove an arbitrary string in Π_{n+1} appears in $n{+}1\ n\ \cdots\ 1 \circ \mathsf{recycle}(S)$ where $S = C(n)$. Let this arbitrary string equal $a_i\ a_{i+1}\ \cdots\ a_n\ n{+}1\ b_1\ b_2\ \cdots b_{i-1}$ for some i satisfying $1 \leq i \leq n{+}1$. We choose this expression for our arbitrary string since we will find α and β such that the following criteria hold

1. α has suffix $a_i\ a_{i+1}\ \cdots\ a_n$, and
2. β has prefix $b_1\ b_2\ \cdots\ b_{i-2}$, and
3. α is followed by β in $\Pi_C(n)$ by applying σ_r, and
4. $\alpha \circ \mathsf{recycle}(r)$ contains the arbitrary string $a_i\ a_{i+1}\ \cdots\ a_n\ n{+}1\ b_1\ b_2\ \cdots b_{i-1}$.

The result is trivial when $n{+}1$ is in the first, last, or second-last position of the arbitrary string. In the remaining cases we define the following

- $\gamma := g_1\ g_2\ ...g_n := b_1\ b_2\ ...\ b_{i-1}\ a_i\ a_{i+1}\ ...\ a_n$ and $p = \lfloor \frac{n}{2} \rfloor - 1$ and $q = \lceil \frac{n}{2} \rceil - 1$,
- x_b is the length of the longest $n\ n{-}1\ n{-}2\ \cdots$ prefix in $g_1\ g_2\ \cdots\ g_{i-2}$,
- y_a is the length of the longest $\cdots\ 3\ 2\ 1$ suffix in $g_i\ g_{i+1}\ \cdots\ g_n$,
- x' is the length of the longest $n\ n{-}1\ n{-}2\ \cdots$ prefix in $g_1\ g_2\ \cdots\ g_p$, and
- y' is the length of the longest $\cdots\ 3\ 2\ 1$ suffix in $g_{n-q+1}\ g_{n-q+2}\ \cdots\ g_n$.

One difference between (x_b, y_a) and (x', y') is that the former considers $b_1\ b_2\ \cdots\ b_{i-2}$ and $a_i\ a_{i+1}\ \cdots\ a_n$ separately, whereas the latter considers γ as a whole. Choose

$$\alpha := \begin{cases}
b_{i-1}\ b_1\ b_2\ ...\ b_{i-2}\ a_i\ a_{i+1}\ ...\ a_n & \text{if } x_b \leq y_a \quad (13a) \\
g_{y'+2}\ g_1\ g_2\ ...\ g_{y'+1}\ g_{y'+3}\ g_{y'+4}\ ...g_n & \text{if } x_b > y_a \text{ and } x' > y' \quad (13b) \\
g_{n-x'}\ g_1\ g_2\ ...\ g_{n-x'-1}\ g_{n-x'+1}\ g_{n-x'+2}\ ...g_n & \text{if } x_b > y_a \text{ and } x' \leq y' \quad (13c)
\end{cases}$$

In each case, we prove the first criterion holds for the choice of α. For (13a) this result is obvious. For (13b) there are two cases two consider. If $x_b \geq x'$, then

$$y' + 3 \leq x' + 2 \leq x_b + 2 \leq i$$

where the inequalities follow from $x' > y'$, $x_b \geq x'$, and $x_b \leq i - 2$, respectively. On the other hand, if $x_b < x'$ then it must be that $y_a = y'$ and so

$$y' + 3 = y_a + 3 \leq x_b + 2 \leq i$$

where the equalities and inequalities follow from $y' = y_a$, $x_b > y_a$, and $x_b \leq i-2$, respectively. In both cases, α has the suffix stated in the first criterion. For (13c) it must be that $i = n - y_a + 1$ and $x_b = x'$. Therefore,

$$n - x' + 1 = n - x_b + 1 \leq n - y_a = i - 1$$

where the equalities and inequalities follow from $x_b = x'$, $x_b < y_a$, and $i = n - y_a + 1$, respectively. Therefore, α has the suffix stated in the first criterion. To complete the proof, use the successor rule from Theorem 5 to verify the remaining criteria. □

Theorem 2 also affirms Conjecture 1 in [7]. That paper uses an equivalent notion of 'recycling' that acts on rotation Gray codes of Π_n instead of Hamilton sequences of $\mathcal{R}_n$. The conjecture is that Corbett's Gray code is 'recyclable' and Theorem 2 equivalently proves that Corbett's Hamilton sequence is 'recyclable'.

7 Open Problems

The following open problems are related to this research:

1. Efficiently generate an explicit Hamilton cycle in $\mathcal{R}_n(\{2, n\})$.
2. Necessary and sufficient conditions for recyclable Hamilton sequences of $\mathcal{R}_n$.
3. A loopless algorithm for generating a recyclable order of Π_n in an array.
4. The diameter of $\mathcal{R}_n(G)$ for $G = \{n-1, n\}$ and $G = \{2, 3, n\}$ and others.

For the fourth problem, we mention that Corbett showed the diameter of $\mathcal{R}_n$ is small [2] and discussed applications of this fact. For the third problem, we mention that there are many loopless algorithms that generate successive permutations in an array, but none are known to be 'recyclable' using the terminology from [7]. In fact, the known recyclable orders using rotations by Corbett [2] and Williams [13] cannot be generated by a loopless array-based algorithm since σ_n cannot be implemented in constant time.

References

1. R.C. Compton and S.G. Williamson, *Doubly Adjacent Gray Codes for the Symmetric Group*, Linear and Multilinear Algebra, 35 3 (1993) 237-293.

2. P. F. Corbett, *Rotator Graphs: An Efficient Topology for Point-to-Point Multiprocessor Networks*, IEEE Transactions on Parallel and Distributed Systems, 3 (1992) 622–626.
3. P. F. Corbett and I.D. Scherson, *Sorting in Mesh Connected Multiprocessors*, IEEE Transactions on Parallel and Distributed Systems, 3 (1992) 626–632.
4. G. Ehrlich, *Loopless Algorithms for Generating Permutations, Combinations and Other Combinatorial Configurations*, IEEE Transactions on Parallel and Distributed Systems, 3 (1992) 626–632. Journal of the ACM, 20 3 (1973) 500–513.
5. Yukihiro Hamada, Feng Bao, Aohan Mei, and Yoshihide Igarashi, *Nonadaptive Fault-Tolerant File Transmission in Rotator Graphs*, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, E79-A 4 (1996) 477–482.
6. A. Holroyd, F. Ruskey, and A. Williams, *Faster Generation of Shorthand Universal Cycles for Permutations*, Proceedings of the 16th Annual International Computing and Combinatorics Conference, COCOON 2010), Nha Trang, Vietnam, July 19-21, LNCS, 6196 (2010) 298–307.
7. A. Holroyd, F. Ruskey, and A. Williams, *Shorthand Universal Cycles for Permutations*, Algorithmica (to appear).
8. D.E. Knuth, *The Art of Computer Programming, Volume 4, Generating All Tuples and Permutations*, Fascicle 2, Addison-Wesley, 2005.
9. Chi-Jung Kuo, Chiun-Chieh Hsu, Hon-Ren Lin, and Kung-Kuei Lin *An Efficient Algorithm for Minimum Feedback Vertex Sets in Rotator Graphs*, Information Processing Letters, 109 9 (2009) 450–453.
10. Hon-Ren Lin and Chiun-Chieh Hsu, *Topological Properties of Bi-Rotator Graphs*, IEICE Transactions on Information and Systems, E86-D 10 (2003) 2172–2178.
11. S. Ponnuswamy and V. Chaudhary, *Embedding of Cycles in Rotator and Incomplete Rotator Graphs*, Proceedings of the Sixth IEEE Symposium on Parallel and Distributed Processing, October 26–29 (1994) 603–610.
12. F. Ruskey and A. Williams, *An Explicit Universal Cycle for the* $(n-1)$*-Permutations of an* n*-set*, ACM Transactions on Algorithms, 6 3 (2010) article 45.
13. A. Williams, *Loopless Generation of Multiset Permutations Using a Constant Number of Variables by Prefix Shifts*, Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2009, New York, NY, USA, January 4–6 (2009) 987–996.
14. Suzuki Yasuto, Kaneko Ken'Ichi, and Nakamori Mario, *Node-Disjoint Paths Problem in a Rotator Graph*, Joho Shori Gakkai Shinpojiumu Ronbunshu, 14 (2003) 93–100.