A Gray Code for Necklaces of Fixed Density

TERRY MINYIH WANG and CARLA D. SAVAGE *
Department of Computer Science
North Carolina State University, Box 8206
Raleigh, NC 27695-8206

January 16, 1997

Abstract

A necklace is an equivalence class of binary strings under rotation. In this paper, we present a Gray code listing of all n-bit necklaces with d ones so that (i) each necklace is listed exactly once by a representative from its equivalence class and (ii) successive representatives, including the last and the first in the list, differ only by the transposition of two bits. The total time required is $O(nN(n,d))$, where $N(n,d)$ denotes the number of n-bit binary necklaces with d ones. This is the first algorithm for generating necklaces of fixed density which is known to achieve this time bound.

1 Introduction

In a combinatorial family, a Gray code is an exhaustive listing of the objects in the family so that successive objects differ only in a small way [Wil]. The classic example is the binary reflected Gray code [Gra], which is a list of all n-bit binary strings in which each string differs from its successor in exactly one bit. By applying the binary Gray code, a variety of problems have been solved and the complexities of the solutions to other problems have been improved [Gar, ChLeDu, ChChCh, Los, Ric]. There are many examples of combinatorial families for which Gray codes are known, including permutations [Joh, Tro], combinations [BuWi, NiWi, Rus1], compositions [Kli], set partitions [Kay], integer partitions [Sav, RaSaWe], binary trees [RuPr, Luc, LuRoRu], and linear extensions [PrRu1, PrRu2, Rus2, Sta, Wes].

When an application requires an exhaustive examination of all objects in a combinatorial family, Gray codes can be used to speed up the task. With a Gray code scheme, it is often possible to list a family of N objects, each of size $O(n)$, in time $O(n + N)$, rather than time $O(n * N)$, by listing the first object, and thereafter listing only the (constant size)

*Research supported by National Science Foundation Grants CCR8906500 and DMS9302505.

change between successive objects. There is an additional advantage if each object has an associated cost, for it is likely that the cost of an object can be computed in constant time from the cost of its predecessor on the Gray code list.

In this paper we consider Gray codes for binary *necklaces.* A *necklace* is an equivalence class of binary strings under rotation. To be precise, let $\Sigma = \{0, 1\}$ and for $n \geq 0$ let Σ^n denote the set of all strings of length n over Σ. Define the rotation operation $\sigma : \Sigma^n \rightarrow \Sigma^n$ by

$$\sigma(x_1 x_2 \cdots x_n) = x_2 \cdots x_n x_1.$$

Then, strings x and y are in the same *necklace* if and only if $\sigma^i(x) = y$ for some integer i. A necklace can be identified by specifying any one of its elements and frequently the lexicographically smallest string is chosen as the representative. The *density* of a necklace is the number of ones in a representative of the necklace. We use $N(n)$ and $N(n, d)$ to denote the number of n-bit necklaces and the number of n-bit necklaces of density d, respectively.

Note that distinct n-bit strings of density d must differ in at least two positions. For this paper, by *a Gray code for necklaces of fixed density* we mean a listing of n-bit binary strings with d ones, exactly one from each necklace, in which successive strings differ in exactly two bit positions. We will show that such a Gray code is always possible and that it gives rise to the most efficient algorithm known for generating necklaces of fixed density.

A simple and elegant algorithm for listing the lexicographically smallest representatives of all n-bit necklaces was given in [FrMa, FrKe] and we refer to this as the FKM algorithm. It was shown in [RuSaWa] that the time required by the FKM algorithm is $\mathrm{O}(N(n))$, that is, constant average time per necklace, which is best possible. The efficiency here is achieved by amortization, rather than a Gray code, since successive representatives listed by the FKM algorithm can differ in $\Omega(n)$ bits. In fact, it can be shown that, in general, there is no listing for n-bit necklaces in which successive representatives differ in just one bit. In such a listing, the density of successive representatives would alternate between even and odd. However, this is impossible for even n, since the numbers of even-density and odd-density necklaces differ by more than one when $n > 0$.

In contrast to the situation with all necklaces, there is no parity problem which precludes, for any n and d, a Gray code for n-bit necklaces of density d. The main result of this paper is to show the existence of a Gray code for any n and d. The proof is constructive and the resulting algorithm, which has been implemented in the programming language **C**, takes time $\mathrm{O}(nN(n, d))$. We note here that the necklace representative used in the Gray code is *not* the lexicographically smallest one, x. Instead, it is the representative obtained from x

by applying σ to x until the leftmost bit is a one.

No previous algorithm, for the problem of listing all n-bit necklaces of fixed density d, is known to be as efficient as $O(nN(n,d))$ for general d. However, in the special case when $n = 2d+1$, a Gray code for n-bit necklaces of density d follows from the Gray code of Ruskey and Proskurowski [RuPr] for balanced parentheses, under a straightforward bijection between these two families and that this algorithm achieves constant average time per object.

In Section 2 we present background and technical lemmas used for the main result. The Gray code construction is presented in Section 3 and its implementation is described and analyzed in Section 4.

2 Background and Technical Lemmas

For $\alpha,\ \beta \in \Sigma^*$, we use $\alpha \leq \beta$ ($\alpha < \beta$) to denote that α precedes (strictly precedes) β in lexicographic order. Let $L(n)$ be the set of lexicographically smallest representatives of the n-bit necklaces. That is,

$$L(n) \;=\; \{x \in \Sigma^n \mid x \leq \sigma^i(x) \ \text{ for } 1 \leq i < n\}.$$

Let $L(n,d)$ be the subset of $L(n)$ of strings of density d. As the backbone of our Gray code construction, we will use a tree of elements of $L(n)$, which was introduced in [WaSa]. For $n \geq 1$, let $\tau\ :\ \Sigma^n \rightarrow \Sigma^n$ be the function

$$\tau(x_1x_2\ldots x_n) \;=\; x_1x_2\ldots\overline{x_n}.$$

Then the tree, $TREE(n)$, is defined recursively by

(i) 0^n is the root of $TREE(n)$ and

(ii) if x is a node of $TREE(n)$, then for $1 \leq i < n$, $\tau\sigma^i(x)$ is a child of x if and only if $\tau\sigma^i(x) \in L(n)$.

As an example, $TREE(7)$ is shown in Figure 1. Note that no element of $L(n)$ appears more than once in $TREE(n)$. For, if $x, y \in L(n)$ and $\tau\sigma^i(x) = \tau\sigma^j(y)$ then $\sigma^i(x) = \sigma^j(y)$ which means x and y are representatives of the same necklace and therefore must be identical. Thus, $TREE(n)$ has no cycles and, since by definition it is connected, it is in fact a tree. It is straightforward to verify that the nodes of $TREE(n)$ are exactly the elements of $L(n)$ and that $L(n,d)$ is the set of nodes on level d. First note that 0^n appears in the tree at level

Figure 1: The tree of 7-bit necklaces, $TREE(7)$.

0 and $0^{n-1}1$ appears at level 1. For $d \geq 2$, assume inductively that all elements in $L(n, d-1)$ appear in $TREE(n)$ at level $d-1$. Then $y \in L(n, d)$ can be written as $y = 0^k \alpha 1 0^i 1$, where $k, i \geq 0$. But then $y = \tau\sigma^{i+1}(0^{k+i+1}\alpha 1)$ and $x = 0^{k+i+1}\alpha 1 \in L(n, d-1)$. By induction, x is in $TREE(n)$ at level $d-1$ and therefore, by definition of $TREE(n)$, y is a child of x at level d.

The following result, crucial to our construction, was proved in [WaSa].

Theorem 1 *For node $x = 0^k 1\alpha$ in $TREE(n)$, with $k \geq 0$ and $\alpha \in \Sigma^*$, and for i satisfying $1 \leq i \leq k$, if $\tau\sigma^i(x) \notin L(n)$, then $\tau\sigma^{i+1} \notin L(n)$.* $\Box$

As a consequence of Theorem 1, if a node $x = 0^k 1\alpha$ in $TREE(n)$ has exactly $c > 0$ children, then those children are $\tau\sigma^1(x), \tau\sigma^2(x), \ldots, \tau\sigma^c(x)$, and none of $\tau\sigma^{c+1}(x), \ldots, \tau\sigma^k(x)$ is in $L(n)$. Thus, if y is a child of $x = 0^k 1\alpha$ in $TREE(n)$, there is a unique i, $1 \leq i \leq k$, such that $y = \tau\sigma^i(x)$.

For node x in $TREE(n)$, let $lev(x)$ denote the level of x in $TREE(n)$, where the root is at level 0. For $d \geq 0$, let $DESC(x, d)$ be the set of descendants of x in $TREE(n)$ on level $d + lev(x)$. A node x is called $d-rich$ if $DESC(x, d) \neq \emptyset$. For example, in Figure 1, $x = 0000001$ is 2-rich, since

$$DESC(0000001, 2) = \{0000111,\ 0001101,\ 0001011,\ 0010101,\ 0010011\} \neq \emptyset$$

Although we are interested in a Gray code for the set $DESC(0^n, d)$, of all n-bit necklaces of density d, we consider only $d \leq \lfloor n/2 \rfloor$, since otherwise, a Gray code for $DESC(0^n, d)$ can be obtained from one for $DESC(0^n, n-d)$ by interchanging the roles of 0 and 1. Our construction will in fact give a Gray code for any set $DESC(x, d)$ with $d + lev(x) \leq \lfloor n/2 \rfloor$.

By repeated application of Theorem 1, if $x = 0^k 1\alpha$, any $z \in DESC(x, d)$ can be written uniquely in the form

$$z = \tau\sigma^{a_d} \ldots \tau\sigma^{a_2}\tau\sigma^{a_1}(x)$$

where $0 < a_i < n$, $\tau\sigma^{a_i} \ldots \tau\sigma^{a_2}\tau\sigma^{a_1}(x) \in L(n)$, and $a_1 + \ldots + a_i \leq k$ for $1 \leq i \leq d$.

For a string $\alpha \in \Sigma^*$, let $max(\alpha)$ be the length of the longest consecutive block of zeros in α. We use repeatedly the facts that since $x \in L(n)$ if and only if it is the lexicographically smallest of all of its rotations, (i) if $0^k 1\alpha \in L(n)$ then $k \geq max(\alpha)$ and (ii) if $k > max(\alpha)$ then $0^k 1\alpha \in L(n)$.

Our Gray code construction depends on the restrictive structure of the necklace tree and requires the technical lemmas below. For a string $\alpha \in \Sigma^*$, define $\#ones(\alpha)$ and $\#zeros(\alpha)$ to be the total number of ones and zeros, respectively, in α. (The density of α is $\#ones(\alpha)$). We denote the *empty string* by λ.

Lemma 1 *Assume that $\alpha = \lambda$ or that $\alpha \in \Sigma^+$ and α ends in 1. For $z, z' \in \Sigma^n$, if $z = 0^a 1\alpha 0^b \beta$ is in $L(n)$, where $a \geq 1$ and $b \geq 1$, then any n-bit string of the form*

$$z' = 0^{a+c} 1\alpha 0^{b-1} 1\beta'$$

must be in $L(n)$ for any $c \geq 0$ and $\beta' \in \Sigma^*$ with $a + c > max(\beta')$. Furthermore, if $z \in L(n, j)$ and $j \leq \lfloor n/2 \rfloor$, then either $a > 1$ or $a \geq \#ones(\beta) - \#zeros(\beta)$.

Proof. Since $z \in L(n)$,

(i)	$a \geq b$ and			
(ii)	either	(a)	$a > max(\alpha)$	
	or	(b)	$a = max(\alpha)$	and
			$z \leq w$	for any rotation w of z.

If (i) and (ii)(a) hold, then $z' \in L(n)$. Otherwise, (i) and ii(b) hold so $a = max(\alpha)$ and $\alpha \neq \lambda$. If $z' \notin L(n)$ then there exist $\alpha_1, \alpha_2 \in \Sigma^*$, with $|\alpha_1| \geq 0$ and $|\alpha_2| \geq 1$, such that α_2 starts with '1' and $\alpha = \alpha_1 0^a \alpha_2$ and z' is greater than its rotation $r(z')$:

$$z' > 0^a \alpha_2 0^{b-1} 1\beta' 0^{a+c} 1\alpha_1 = r(z'). \quad (1)$$

It must follow that $c = 0$. Let $r(z)$ be the rotation of z:

$$r(z) = 0^a \alpha_2 0^b \beta 0^a 1 \alpha_1.$$

Note that $0^{a+c}1\alpha \leq z$, and by ii(b), $z \leq r(z)$. Also, $r(z) < r(z')$, so $0^{a+c}1\alpha < r(z')$. Combining this with (1), since $0^{a+c}1\alpha$ is a prefix of z', so also $0^{a+c}1\alpha$ is a prefix of $r(z')$. However, note that the prefix $0^a\alpha_2 0^{b-1}1$ of $r(z')$ is shorter than $0^{a+c}1\alpha$:

$$\begin{array}{rcll} |0^a\alpha_2 0^{b-1}1| & = & a + |\alpha_2| + b & \\ & \leq & a + |\alpha_2| + a & \text{(by (i))}, \\ & < & a + c + 1 + |\alpha_1| + a + |\alpha_2| & (\text{since} \quad c \geq 0 \quad \text{and} \quad |\alpha_1| \geq 0), \\ & = & |0^{a+c}1\alpha|. & \end{array}$$

Therefore, $0^a\alpha_2 0^{b-1}1$ is a prefix of $0^{a+c}1\alpha$. Thus,

$$0^a\alpha_2 0^{b-1}1 \ \leq \ 0^{a+c}1\alpha \ \leq \ z \leq \ r(z) \ = \ 0^a\alpha_2 0^b \beta 0^a 1 \alpha_1$$

a contradiction.

If $z \in L(n, j)$ and $j \leq \lfloor n/2 \rfloor$, then $\#zeros(z) \geq \#ones(z)$, so

$$a + b + \#zeros(\alpha) + \#zeros(\beta) \ \geq \ 1 + \#ones(\alpha) + \#ones(\beta).$$

Thus,

$$a \ \geq \ (\#ones(\alpha) - \#zeros(\alpha)) \ + \ (\#ones(\beta) - \#zeros(\beta)) \ + \ 1 \ - \ b. \qquad (2)$$

If $a \leq 1$, then since $a \geq b \geq 1$, it follows that $a = b = 1$, so $max(\alpha) \leq 1$. Therefore α cannot have more zeros than ones, since α ends in 1. Thus, from (2), $a \geq \#ones(\beta) - \#zeros(\beta)$.
□

We introduce a notation to label key nodes of $TREE(n)$ used in the Gray code construction. For a binary string, z, define u, u', u'', v, v', and w as follows:

when $d \geq 0$,
$$u(z, d) = (\tau\sigma)^d(z);$$
when $d = 0$,
$$v(z, d) = z;$$
when $d \geq 1$,
$$\begin{array}{rcl} v(z, d) & = & \tau\sigma^2(\tau\sigma)^{d-1}(z), \\ w(z, d) & = & \tau\sigma^3(\tau\sigma)^{d-1}(z), \\ u'(z, d) & = & (\tau\sigma)^{d-1}\tau\sigma^2(z), \\ u''(z, d) & = & (\tau\sigma)^{d-1}\tau\sigma^3(z); \end{array}$$
when $d \geq 2$,
$$v'(z, d) = \tau\sigma^2(\tau\sigma)^{d-2}\tau\sigma^2(z).$$

a. When $d = 1$

b. When $d \geq 2$

Figure 2: Labeling key tree nodes.

For example, in Figure 1, if $z = 0000001$, then $u(z,2) = 0000111$, $v(z,2) = 0001101$, $u'(z,2) = 0001011$, $v'(z,2) = 0010101$, and $u''(z,2) = 0010011$. Note that for general z, $u(z,0) = z$, $u'(z,1) = v(z,1)$ and $u''(z,1) = w(z,1)$.

Let $x \in L(n)$ and let $y_i = \tau\sigma^i(x)$ denote the i-the child of x. Further use of the vertex labels is illustrated in Figure 2. The lemma below shows that existence of a node at certain locations in $TREE(n)$ forces the existence of nodes at certain other locations in $TREE(n)$.

Lemma 2 *Let $x \neq 0^n$ be a node in $TREE(n)$. Let $y_i = \tau\sigma^i(x)$ and assume that $j \leq \lfloor n/2 \rfloor$. For*

$$\begin{array}{llllll}
1 \leq i < n-1,\ d \geq 1, & (i) & \text{if} & u(y_{i+1},d) \in L(n,j), & \text{then} & u(y_i,d) \in L(n,j), \\
 & (ii) & \text{if} & u(y_{i+1},d) \in L(n,j), & \text{then} & v(y_i,d) \in L(n,j), \\
 & (iii) & \text{if} & u(y_{i+1},d) \in L(n,j), & \text{then} & u'(y_i,d) \in L(n,j); \\
1 \leq i < n-1,\ d \geq 2, & (iv) & \text{if} & v(y_{i+1},d) \in L(n,j), & \text{then} & v'(y_i,d) \in L(n,j); \\
1 \leq i < n-2,\ d \geq 1, & (v) & \text{if} & v(y_{i+2},d) \in L(n,j), & \text{then} & u''(y_i,d) \in L(n,j).
\end{array}$$

Proof. (See Figure 2.) Write x as $x = 0^k 1\alpha$ where $\alpha \in \Sigma^*$. Then,

$$\begin{array}{lclcl}
u(y_{i+1},d) & = & (\tau\sigma)^d \tau\sigma^{i+1}(x) & = & 0^{k-i-d-1}1\alpha 0^i 1^{d+1}, \\
u(y_i,d) & = & (\tau\sigma)^d \tau\sigma^i(x) & = & 0^{k-i-d}1\alpha 0^{i-1} 1^{d+1}, \\
v(y_i,d) & = & \tau\sigma^2(\tau\sigma)^{d-1}\tau\sigma^i(x) & = & 0^{k-i-d-1}1\alpha 0^{i-1} 1^d 01, \\
u'(y_i,d) & = & (\tau\sigma)^{d-1}\tau\sigma^2\tau\sigma^i(x) & = & 0^{k-i-d-1}1\alpha 0^{i-1} 101^d, \\
v(y_{i+1},d) & = & \tau\sigma^2(\tau\sigma)^{d-1}\tau\sigma^{i+1}(x) & = & 0^{k-i-d-2}1\alpha 0^i 1^d 01, \\
v'(y_i,d) & = & \tau\sigma^2(\tau\sigma)^{d-2}\tau\sigma^2\tau\sigma^i(x) & = & 0^{k-i-d-2}1\alpha 0^{i-1} 101^{d-1}01, \\
v(y_{i+2},d) & = & \tau\sigma^2(\tau\sigma)^{d-1}\tau\sigma^{i+2}(x) & = & 0^{k-i-d-3}1\alpha 0^{i+1} 1^d 01, \\
u''(y_i,d) & = & (\tau\sigma)^{d-1}\tau\sigma^3\tau\sigma^i(x) & = & 0^{k-i-d-2}1\alpha 0^{i-1} 1001^d.
\end{array}$$

Cases (i) - (iii) of the lemma follow from Lemma 1 with $a = k-i-d-1$, $b = i$, $\beta = 1^{d+1}$, and

$$\begin{array}{ll}
\text{for (i), with} & c = 1,\ \beta' = 1^d, \\
\text{for (ii), with} & c = 0,\ \beta' = 1^{d-1}01, \\
\text{for (iii), with} & c = 0,\ \beta' = 01^d.
\end{array}$$

In each case (i) - (iii), $max(\beta') \leq 1$. Note that by Lemma 1 either $a > 1$, so that $a + c > 1 + c > 1 - c = max(\beta')$, or

$$a + c \geq \#ones(\beta) - \#zeros(\beta) + c = d + 1 + c \geq 2 + c \geq 2 > max(\beta').$$

Case (iv) follows from Lemma 1 with $a = k-i-d-2$, $b = i$, $\beta = 1^d01$, $c = 0$, and $\beta' = 01^{d-1}01$. Note that $d \geq 2$ in this case and by Lemma 1, either $a > 1 = max(\beta')$, or $a \geq \#ones(\beta) - \#zeros(\beta) = d \geq 2 > max(\beta')$.

Case (v) follows from Lemma 1 with $a = k-i-d-3$, $b = i$, $\beta = 01^d01$, $c = 1$, and $\beta' = 001^d$. Note that $a \geq i+1 \geq 2$, since $v(y_{i+2},d) \in L(n)$. Thus $a + c = a + 1 > 2 = max(\beta')$. □

Let x be a node in $TREE(n)$ with c children $y_1, \ldots y_c$, where $y_j = \tau\sigma^j(x)$. For $d \geq 1$ and $1 \leq i \leq c$, define

$$V(x,d,i) \ = \ \bigcup_{j=i}^{c} DESC(y_j, d-1).$$

Then note that $DESC(x,d) = V(x,d,1)$. In Figure 1,

$V(0000001,2,2) = DESC(0000101,1) \cup DESC(0001001,1) = \{0001011,\ 0010101,\ 0010011\}$.

The next section gives a recursive construction of a Gray code for $V(x,d,i)$, based on the following recursive decomposition of $V(x,d,i)$:

$$V(x,d,i) \ = \ DESC(y_i, d-1) \cup V(x,d,i+1).$$

Note that for a given $d \geq 1$, not every child of x need be $(d-1)$-rich. Let $r(x,d)$ be the number of $(d-1) - rich$ children of x. Corollary 1(a) below generalizes Theorem 1 to establish that the $(d-1)$-rich children of x must be exactly $y_1, y_2, \ldots, y_{r(x,d)}$.

Corollary 1 *Let $x \in L(n,b)$, $b \geq 1$, $d \geq 1$, $b + d \leq \lfloor n/2 \rfloor$. For $1 \leq i \leq n$, let $y_i = \tau\sigma^i(x)$.*

(a) If $V(x,d,i) \neq \emptyset$, then $u(y_i, d-1) \in DESC(y_i, d-1)$.

(b) If $|V(x,d,i)| \geq 2$, then $DESC(y_i, d-1)$ contains both $u(y_i, d-1)$ and $v(y_i, d-1)$.

Proof. Induction on d. If $d = 1$, the result follows from Theorem 1. Assume the result is true for d' satisfying $1 \leq d' < d$. If $V(x,d,i) \neq \emptyset$, then $DESC(y_{i+j}, d-1) \neq \emptyset$ for some j satisfying $0 \leq j < n - i$ and $y_{i+j} \in L(n)$.

Let $z = \tau\sigma(y_{i+j})$. Then $u(z, d-2) = u(y_{i+j}, d-1)$. Since $DESC(y_{i+j}, d-1) = V(y_{i+j}, d-1, 1)$, by induction, $u(z,d-2) \in DESC(z, d-2) \subseteq DESC(y_{i+j}, d-1)$ and therefore $u(y_{i+j}, d-1) \in DESC(y_{i+j}, d-1)$. So by Lemma 2(i), $u(y_{i+j-1}, d-1) \in DESC(y_{i+j-1}, d-1)$. Part (a) follows from $j-1$ further applications of Lemma 2(i). Part (b) follows from (a) and Lemma 2(ii). □

3 The Gray Code Construction

We first introduce a different collection of necklace representatives which are a slight variation on the L-representatives. Note that although the L-representatives $x = 00000011101$ and $y = 00001110001$ differ in four bit positions, there exist rotations x', y' of x, y, respectively, which differ in only two bit positions. Although we could call x and y "adjacent" if there exist rotations of x and y which differ in only two bit positions, we found that by using the M-representatives, defined below, we could make explicit the rotations of x and y which differ in only two bit positions, at least for the adjacencies which will be used in our construction.

For a necklace representative $x \in L(n)$, either $x = 0^n$ or x can be written uniquely as $x = 0^k 1\alpha$ for some $k \geq 0$ and $\alpha \in \Sigma^*$. Define a function M on $L(n)$ by

$$M[x] = \begin{cases} x & \text{if } x = 0^n \\ 1\alpha 0^k & \text{if } x = 0^k 1\alpha. \end{cases}$$

Since $M[x]$ and x are in the same necklace, the set

$$\{M[x] \mid x \in L(x)\}$$

is an alternate set of necklace representatives, called *M-representatives.*

Define $x,\ y \in L(n)$ to be *M-adjacent* if $M[x]$ and $M[y]$ differ only by the interchange of a 0 and a 1. Let G_n be the graph whose vertex set is $L(n)$, with two vertices adjacent in G_n if and only if they are M-adjacent. Let $G_n[x,d,i]$ be the subgraph of G_n induced by $V(x,d,i)$. Note that if $z_1, z_2, \ldots, z_s$ is a hamilton path in $G_n[x,d,i]$, then $M[z_1], M[z_2], \ldots, M[z_s]$ is a Gray code for the necklaces in $V(x,d,i)$. For example, in Figure 1, the graph $G_7[0000001,2,1]$ is a graph with vertex set

$$V(0000001,2,1) = \{0000111,\ 0001101,\ 0001011,\ 0010101,\ 0010011\}.$$

A hamilton path in this graph is:

$$0000111,\ 0010101,\ 0001011,\ 0010011,\ 0001101$$

and the corresponding list of M-representatives, giving a Gray code, is:

$$1110000,\ 1010100,\ 1011000,\ 1001100,\ 1101000.$$

Our goal is to find a Gray code for the necklaces in $DESC(0^n,d) = V(0^n,d,1) = V(0^{n-1}1, d-1, 1)$ when $1 \leq d \leq \lfloor n/2 \rfloor$. This will follow if we can show that $G_n[x,d,i]$ has a hamilton path for every (x,d,i) satisfying $x \in L(n)$, $lev(x)+d \leq \lfloor n/2 \rfloor$, and $1 \leq i \leq r(x,d)$.

The strategy will be recursive. Let $y_i = \tau\sigma^i(x)$. Since it was shown in Section 2 that $V(x,d,i) = V(y_i, d-1, 1) \cup V(x,d,i+1)$, hamilton paths/cycles recursively constructed in $G_n[y_i, d-1, 1]$ and $G_n[x,d,i+1]$ will be linked to form a hamilton path/cycle in $G_n[x,d,i]$. The full construction is contained in Theorem 2 at the end of this section, following some further technical lemmas.

In the next sequence of lemmas, we examine the structure of $G_n[x,d,i]$. In particular, we show that the graph is complete whenever $d=1$ or $i=r(x,d)$. In addition, we establish the existence of certain edges which will be used to link together hamilton cycles recursively constructed. When x is fixed, we let $y_i = \tau\sigma^i(x)$.

Lemma 3 *For any node x in $TREE(n)$, $G_n[x,1,1]$ is complete.*

Proof. If $x=0^n$, the only child of x is $0^{n-1}1$. Otherwise, x has the form $x=0^k1\alpha$ for some $\alpha \in \Sigma^*$. Let $\tau\sigma^i(x)$ and $\tau\sigma^j(x)$ be distinct children of x with $1 \le i,j \le n$. Then

$$M[\tau\sigma^i(x)] \;=\; 1\alpha 0^{i-1}10^{k-i}$$

and

$$M[\tau\sigma^j(x)] \;=\; 1\alpha 0^{j-1}10^{k-j},$$

and these differ only by the exchange of a 0 and 1. □

Lemma 4 *For $x \ne 0^n$ in $TREE(n)$ and $d \ge 2$, let $r=r(x,d)$. Let $y_i=\tau\sigma^i(x)$. Any z in $DESC(y_r,d-1)$ has the form*

$$z = \tau\sigma^{a_{d-1}}\tau\sigma^{a_{d-2}}\cdots\tau\sigma^{a_1}(y_r)$$

where $1 \le a_i \le 2$ for $1 \le i \le d-1$ and $a_i = 2$ for at most one $i \in \{1,\cdots,d-1\}$.

Proof. Since $x \ne 0^n$, x can be written as $0^k1\alpha$ for some $\alpha \in \Sigma^*$ where $\alpha=\lambda$ or α ends in 1. Any $z \in DESC(y_r,d-1)$ can be written uniquely as $z=\tau\sigma^{a_{d-1}}\ldots\tau\sigma^{a_1}(y_r)$ where $1 \le a_i \le n$ for $1 \le i \le d-1$. We show that $a_1+\cdots+a_{d-1} \le d$.

By definition of $r=r(x,d)$, $z' = u(y_{r+1},d-1) \notin L(n)$. Expand z and z' as

$$z = \tau\sigma^{a_{d-1}}\cdots\tau\sigma^{a_1}\tau\sigma^r(x) = 0^{k-r-a_1-\cdots-a_{d-1}}1\alpha 0^{r-1}10^{a_1-1}1\cdots 0^{a_{d-1}-1}1,$$

$$z' = u(y_{r+1},d-1) = (\tau\sigma)^{d-1}\tau\sigma^{r+1}(x) = 0^{k-r-d}1\alpha 0^r 1^d.$$

Let $s=a_1+\cdots+a_{d-1}$ and assume that $s>d$. Since $z \in L(n)$, $k-r-s \;\ge\; max(\alpha)$ and

$$k-r-d \;>\; k-r-d-1 \;\ge\; k-r-s \;\ge\; r-1. \tag{3}$$

But then since $z' \notin L(n)$, it must be that $k-r-d=r$ and z' is larger than its rotation $r(z')$:

$$0^r1\alpha 0^r 1^d \;=\; z' \;>\; r(z') \;=\; 0^r1^d0^r1\alpha. \tag{4}$$

Let $\beta = 0^{a_1 - 1}1 \ldots 0^{a_{d-1}-1}1$ and let $r(z)$ be the rotation of $z = 0^{k-r-s}1\alpha 0^{r-1}1\beta$:

$$r(z) \ = \ 0^{r-1}1\beta 0^{k-r-s}1\alpha.$$

Since $|\beta| = s > d$ and $\#ones(\beta) = d-1$,

$$\beta 0^{r-1}1\alpha \ < \ 1^{d-1}0^r 1\alpha \tag{5}$$

But then,

$$\begin{array}{rcll} 0z & = & 0^{k-r-s+1}1\alpha 0^{r-1}1\beta & \\ & \geq & 0^{k-r-d}1\alpha 0^{r-1}1\beta & \text{since} \quad s > d, \\ & = & 0^r 1\alpha 0^{r-1}1\beta & \text{since} \quad k-r-d = r, \\ & > & z' > r(z') & \text{by (4)} \\ & = & 0^r 1^d 0^r 1\alpha & \\ & > & 0^r 1\beta 0^{r-1}1\alpha & \text{by (5)} \\ & \geq & 0^r 1\beta 0^{k-r-s}1\alpha & \text{by (3)} \\ & = & 0r(z). & \end{array}$$

Thus $0z > 0r(z)$, so $z > r(z)$ which is impossible since z is a necklace. □

Lemma 5 *For $x \neq 0^n$ in $TREE(n)$ and $d \geq 2$, let $r = r(x,d)$. Then, $G_n[x,d,r]$ is complete.*

Proof. Since $x \neq 0^n$, x can be written as $x = 0^k 1\alpha$ where $\alpha \in \Sigma^*$ and either $\alpha = \lambda$ or α ends in 1. By Lemma 4, any vertex of $G_n[x,d,r] = G_n[y_r, d-1, 1]$ has the form

$$\tau\sigma^{a_{d-1}} \ldots \tau\sigma^{a_1}\tau\sigma^r(x) \ = \ 0^{k-r-a_1-a_2-\cdots-a_{d-1}}1\alpha 0^{r-1}10^{a_1-1}1\ldots 0^{a_{d-1}-1}1,$$

where $1 \leq a_i \leq 2$ for $1 \leq i \leq d-1$ and $a_i = 2$ for at most one $i \in \{1, \ldots, d-1\}$. This means that $s = a_1 + \cdots + a_{d-1}$ is either $d-1$ or d and any vertex has the form

$$\text{(i) } 0^{k-r-d+1}1\alpha 0^{r-1}1^d \qquad (\text{if} \quad s = d-1)$$

or

$$\text{(ii) } 0^{k-r-d}1\alpha 0^{r-1}1^i 01^{d-i} \qquad (\text{if} \quad s = d.)$$

Clearly, any two vertices of this form are M-adjacent. □

Figure 3: Edges in G_n, when $d = 1$, labeled by corresponding part of Lemma 6.

Figure 4: Edges in G_n, when $d \geq 2$, labeled by corresponding part of Lemma 6.

Lemma 6 *Let $x \neq 0^n$ be a node in $TREE(n)$ with $d \geq 1$ and $r = r(x,d)$. For each of the following pairs of strings z, z', if both z and z' are in $L(n)$, then they are M-adjacent:*

(a)	$u(y_i,d)$	*and*	$v(y_i,d)$	*for*	$d \geq 1$	*and*	$1 \leq i \leq r$
(b)	$u(y_i,d)$	*and*	$u(y_{i+1},d)$	*for*	$d \geq 1$	*and*	$1 \leq i < r$
(c)	$u(y_i,d)$	*and*	$v(y_{i+1},d)$	*for*	$d \geq 1$	*and*	$1 \leq i < r$
(d)	$v(y_i,d)$	*and*	$u(y_{i+1},d)$	*for*	$d \geq 1$	*and*	$1 \leq i < r$
(e)	$u'(y_i,d)$	*and*	$u(y_{i+1},d)$	*for*	$d \geq 2$	*and*	$1 \leq i < r$
(f)	$v'(y_i,d)$	*and*	$v(y_{i+1},d)$	*for*	$d \geq 2$	*and*	$1 \leq i < r$
(g)	$u'(y_i,d)$	*and*	$u(y_{i+2},d)$	*for*	$d \geq 2$	*and*	$1 \leq i < r-1$
(h)	$v'(y_i,d)$	*and*	$u(y_{i+2},d)$	*for*	$d \geq 2$	*and*	$1 \leq i < r-1$
(i)	$w(y_i,1)$	*and*	$v(y_{i+1},1)$	*for*	$1 \leq i < r$		
(j)	$v(y_i,1)$	*and*	$u(y_{i+2},1)$	*for*	$1 \leq i < r-1$		
(k)	$w(y_i,1)$	*and*	$u(y_{i+2},1)$	*for*	$1 \leq i < r-1$.		

Proof. These edges are illustrated in Figure 3, when $d = 1$, and Figure 4, when $d > 1$. Using the definitions of the vertices and the fact that x has the form $0^k1\alpha$, for $k > 0$ and $\alpha \in \Sigma^*$, the lemma can be verified directly for each pair (a) - (k). For example, in (c),

$$
\begin{aligned}
M[u(y_i,d)] &= M[(\tau\sigma)^d(y_i)] \\
&= M[(\tau\sigma)^d\tau\sigma^i(x)] \\
&= M[(\tau\sigma)^d\tau\sigma^i(0^k1\alpha)] \\
&= M[0^{k-i-d}1\alpha0^{i-1}1^{d+1}] \\
&= 1\alpha0^{i-1}1^{d+1}0^{k-i-d}
\end{aligned}
$$

and similarly,

$$
\begin{aligned}
M[v(y_{i+1},d)] &= M[\tau\sigma^2(\tau\sigma)^{d-1}(y_{i+1})] \\
&= M[\tau\sigma^2(\tau\sigma)^{d-1}\tau\sigma^{i+1}(0^k1\alpha)] \\
&= M[0^{k-i-d-2}1\alpha0^i1^d01] \\
&= 1\alpha0^i1^d010^{k-i-d-2}.
\end{aligned}
$$

Exchanging the last 1 in $M[v(y_{i+1},d)]$ with the last 0 in the block of i zeros gives $M[u(y_i,d)]$. □

The following two lemmas describe strategies which will be used repeatedly in Theorem 2 for linking together hamilton paths/cycles in $G_n[y_i, d-1, 1]$ and $G_n[x, d, i+1]$ to get a hamilton path/cycle in $G_n[x,d,i]$. Although linking is straightforward in the general case, many cases require special attention if d is small or if i is close to $r(x,d)$.

First, define hamilton paths and cycles of special types in $G_n[x,d,i]$ as follows:

$O(x,d,i)$ cycle: a hamilton cycle containing edges
$u(y_i, d-1)v(y_i, d-1)$ and $u(y_i, d-1)v(y_{i+1}, d-1)$

$E(x,d,i)$ cycle: a hamilton cycle containing edges
$u(y_i, d-1)v(y_i, d-1)$ and $u(y_i, d-1)u(y_{i+1}, d-1)$

$UV(x,d,i)$ path: a hamilton path from $u(y_i, d-1)$ to $v(y_i, d-1)$

$P(x,d,i)$ path: a hamilton path from $v(y_i, d-1)$ to $u(y_{i+1}, d-1)$

$Q(x,d,i)$ path: a hamilton path from $u(y_i, d-1)$ to $u(y_{i+1}, d-1)$

Note that $O(x,d,i)$ and $E(x,d,i)$ cycles contain $UV(x,d,i)$ paths. The O and E type hamilton cycles are so named because for large enough d, it turns out that $G_n[x,d,i]$ has an O type cycle for odd d and an E type cycle for even d. The UV, P, and Q type hamilton paths are actually cycles, since by Lemma 6 their start and end vertices are adjacent. However, they will be used only as paths for splicing into O and E type cycles.

The construction of the main theorem splits $G_n[x,d,i]$ into two subgraphs, one of which contains an O or E type cycle and the other contains a hamilton path of type UV, P, or Q. An O type cycle for $G_n[x,d,i]$ is then formed by splicing together a cycle and path recursively constructed in the subproblems: either an E type cycle plus a UV path or an E type cycle plus a P type path. Similarly, An E type cycle for $G_n[x,d,i]$ can be formed by splicing together an O type cycle plus a UV path or an O type cycle plus a Q type path. The details are given in Lemmas 7 and 8 below.

Lemma 7 *For $x \in L(n,b)$ with $b, d \geq 1$ and $b+d \leq \lfloor n/2 \rfloor$, assume that $r(x,d) \geq 2$ and let y_i be the i-th child of x. For $i < r(x,d)$, if $G_n[y_i, d-1, 1]$ has an $E(y_i, d-1, 1)$ (respectively $O(y_i, d-1, 1)$) cycle, then $G_n[x,d,i]$ has an $O(x,d,i)$ (respectively $E(x,d,i)$) cycle as long as $G_n[x,d,i+1]$ has:*

(A) a $UV(x,d,i+1)$ path or

(B) both $P(x,d,i+1)$ and $Q(x,d,i+1)$ paths.

Proof. (See Figure 5.) Let C be an $E(y_i, d-1, 1)$ cycle in $G_n[y_i, d-1, 1]$ and for $1 \leq j \leq r(y_i, 1)$, let z_j be the j-th child of y_i. Then C contains the edges $e_1 = u(z_1, d-2)v(z_1, d-2) = u(y_i, d-1)v(y_i, d-1)$ and $e_2 = u(z_1, d-2)u(z_2, d-2) = u(y_i, d-1)u'(y_i, d-1)$. We seek a hamilton cycle in $G_n[x,d,i]$ containing edge e_1 and the edge $e_3 = u(y_i, d-1)v(y_{i+1}, d-1)$, which edge exists by Lemma 6(c).

Figure 5: Construction of $O(x, d, i)$ and $E(x, d, i)$ cycles in Lemma 7.

Let R_A be a $UV(x,d,i+1)$ path in $G_n[x,d,i+1]$ from $u(y_{i+1},d-1)$ to $v(y_{i+1},d-1)$. Note that $e_4 = u'(y_i,d-1)u(y_{i+1},d-1)$ is an edge by Lemma 6(e). Then $(C-e_2)+e_3+e_4+R_A$ is a hamilton cycle in $G_n[x,d,i]$ containing edges e_1 and e_3.

Let R_B be a $P(x,d,i+1)$ path in $G_n[x,d,i+1]$ from $v(y_{i+1},d-1)$ to $u(y_{i+2},d-1)$ and let $e_5 = u'(y_i,d-1)u(y_{i+2},d-1)$, which is an edge by Lemma 6(g). Then $(C-e_2)+e_3+e_5+R_B$ is a hamilton cycle in $G_n[x,d,i]$ containing edges e_1 and e_3.

On the other hand, if C' is an $O(y_i,d-1,1)$ cycle in $G_n[y_i,d-1,1]$, it contains edge e_1 as well as e_2' where $e_2' = u(z_1,d-2)v(z_2,d-2) = u(y_i,d-1)v'(y_i,d-1)$. We seek a hamilton cycle in $G_n[x,d,i]$ containing edge e_1 and the edge $e_3' = u(y_i,d-1)u(y_{i+1},d-1)$, which edge exists by Lemma 6(b).

With R_A as before, note that $e_4' = v'(y_i,d-1)v(y_{i+1},d-1)$ is an edge by Lemma 6(f). Then $(C'-e_2)+e_3'+e_4'+R_A$ is the required hamilton cycle in $G_n[x,d,i]$.

Let R_B' be a $Q(x,d,i+1)$ path in $G_n[x,d,i+1]$ from $u(y_{i+1},d-1)$ to $u(y_{i+2},d-1)$ and let $e_5' = v'(y_i,d-1)u(y_{i+2},d-1)$, which is an edge by Lemma 6(h). Then $(C-e_2')+e_3'+e_5'+R_B'$ is the required hamilton cycle in $G_n[x,d,i]$. □

A graph is *trivial* if it has only one vertex.

Lemma 8 *For $x \in L(n,b)$ with $b,d \geq 1$ and $b+d \leq \lfloor n/2 \rfloor$, assume that $r(x,d) \geq 2$ and let $r = r(x,d)$. If $G_n[x,d,r]$ is trivial, then $G_n[x,d,r-1]$ has both $P(x,d,r-1)$ and $Q(x,d,r-1)$ paths as long as $G_n[y_{r-1},d-1,1]$ has either*

(A) a $UV(y_{r-1},d-1,1)$ path or

(B) both $P(y_{r-1},d-1,1)$ and $Q(y_{r-1},d-1,1)$ paths.

Proof. (See Figure 6.) If $G_n[x,d,r]$ is trivial, it contains only the vertex $u(y_r,d-1)$, by Corollary 1(a). By Lemma 6(d) and (b), respectively, $G_n[x,d,r-1]$ contains the edges $e_1 = v(y_{r-1},d-1,1)u(y_r,d-1,1)$ and $e_2 = u(y_{r-1},d-1,1)u(y_r,d-1,1)$, as long as the endpoints exist. It suffices to show that if (A) or (B) holds, $G_n[x,d,r-1]$ has a hamilton cycle C_1 containing e_1 and a hamilton cycle C_2 containing e_2.

If (A) holds, let H be a $UV(y_{r-1},d-1,1)$ path in $G_n[y_{r-1},d,1]$ from $u(y_{r-1},d-1)$ to $v(y_{r-1},d-1)$. Then $C_1 = C_2 = H + e_1 + e_2$ gives both required cycles in $G_n[x,d,r-1]$.

If (B) holds, let z_j be the j-th child of y_{r-1} and let H_1 and H_2 be hamilton paths in $G_n[y_{r-1},d,1]$ from $v(z_1,d-2) = v(y_{r-1},d-1)$ to $u(z_2,d-2) = u'(y_{r-1},d-1)$ and from $u(z_1,d-2) = u(y_{r-1},d-1)$ to $u(z_2,d-2) = u'(y_{r-1},d-1)$, respectively. Let e_3 be the

Figure 6: Construction of $P(x, d, r-1)$ and $Q(x, d, r-1)$ paths in Lemma 8.

edge $e_3 = u'(y_{r-1}, d-1)u(y_r, d-1)$ which exists by Lemma 6(e). Then the required cycles in $G_n[x, d, r-1]$ are $C_1 = H_1 + e_1 + e_3$ and $C_2 = H_2 + e_2 + e_3$. □

We can now prove the main result of the paper.

Theorem 2 *For $x \in L(n, b)$ with $b \geq 1$, let d and i be integers such that $1 \leq d + b \leq \lfloor n/2 \rfloor$ and $1 \leq i \leq r(x, d)$. Then $G_n[x, d, i]$ has a hamilton cycle whenever $|V(x, d, i)| \geq 3$.*

Proof. We prove the following claim under the hypotheses of the theorem.

CLAIM: Assume $|V(x, d, i)| \geq 2$.

[Exception A Subclaim:]
If

(i) $d = 1$, or

(ii) $i = r(x, d)$, or

(iii) $d = 2$ and $V(x, d, i) = \{u(y_i, 1),\ v(y_i, 1),\ u(y_{i+1}, 1),\ v(y_{i+1}, 1)\}$.

then $G_n[x, d, i]$ has a $UV(x, d, i)$ path.

[Exception B Subclaim:]
Otherwise, if $i + 1 = r(x, d)$ and $G_n[x, d, i+1]$ is trivial, then $G_n[x, d, i]$ has both $P(x, d, i)$ and $Q(x, d, i)$ paths.

[General Case:]
Otherwise, $G_n[x, d, i]$ has an $E(x, d, i)$ cycle when d is even and an $O(x, d, i)$ cycle when d is odd.

Note that the theorem follows from the CLAIM, since the origin and terminus of any UV, P, or Q path are adjacent by Lemma 6.

Proof of CLAIM: Let $r = r(x, d)$. The proof is by induction on d and $r - i$. If $d = 1$, then $G_n[x, d, i]$ is complete since it is a subgraph of $G_n[x, 1, 1]$, which is complete by Lemma 3. Similarly, when $r - i = 0$ and $d \geq 2$, $G_n[x, d, i]$, is complete by Lemma 5. In both cases, by Corollary 1(b), $G_n[x, d, i]$ contains $u(y_i, d-1)$ and $v(y_i, d-1)$, and, since it is complete, it contains a hamilton path joining these two vertices. This satisfies Subclaim A(i, ii) of the CLAIM.

Assume inductively that when $d \geq 2$ and $r - i \geq 1$, the CLAIM is true for all (d', i') with both $d' = d - 1$ and $r - i' \geq 0$ or both $d' = d$ and $r - i' \leq r - i - 1$. We show the claim is true for (d, i).

Since $r - i \geq 1$, $u(y_{i+1}, d-1) \in L(n)$, so by Lemma 2(iii), $u'(y_i, d-1) \in L(n)$. Thus, $r(y_i, d-1) \geq 2$ and by Corollary 1(b), $G_n[y_i, d-1, 1]$ contains $u(y_i, d-1)$ and $v(y_i, d-1)$, as well as $u'(y_i, d-1)$.

<u>Case $d = 2$</u>:

If $d = 2$ and $r - i \geq 1$, by induction, $G_n[y_i, 1, 1]$ has a $UV(y_i, 1, 1)$ path H from $u(y_i, 1)$ to $v(y_i, 1)$. By Lemma 6(a,b), $G_n[x, 2, i]$ contains the edges $e_1 = u(y_i, 1)$ to $v(y_i, 1)$ and $e_2 = u(y_i, 1)$ to $u(y_{i+1}, 1)$

If $G_n[x, 2, i+1]$ is trivial (Exception B), let e_3 be the edge $e_3 = v(y_i, 1)$ to $u(y_{i+1}, 1)$, which exists by Lemma 6(d). Then $H + e_2 + e_3$ is a hamilton cycle in $G_n[x, 2, i]$ containing both $P(x, 2, i)$ and $Q(x, 2, i)$ paths, as required for Exception B. (See Figure 7(a).)

If $G_n[x, 2, i]$ is Exception A(iii) of the CLAIM, edges $u(y_{i+1}, 1)v(y_{i+1}, 1)$ and $u(y_i, 1)v(y_{i+1}, 1)$ exist by Lemma 6(a,c). These two edges together with e_3 give a $UV(x, 2, i)$ path from $u(y_i, 1)$ to $v(y_i, 1)$, as required. (See Figure 7(b)).

Otherwise, $G_n[x, 2, i]$ is none of the exceptions, so, since $d = 2$ is even, we must show it has an $E(x, 2, i)$ cycle containing e_1 and e_2. By induction, $G_n[x, 2, i+1]$ has either a hamilton path R_1 from $u(y_{i+1}, d-1)$ to $u(y_{i+2}, d-1)$ (Exception B) or a hamilton path R_2 from $u(y_{i+1}, d-1)$ to $v(y_{i+1}, d-1)$ (Exception A and General Case).

If $r(y_i, 1) = 2$, then by Lemma 2(v), $v(y_{i+2}, 1) \notin L(n)$ and therefore $G_n[x, 2, i+1]$ is Exception B, so it contains R_1. In this case, let e_4 be the edge $v(y_i, 1)\ u(y_{i+2}, 1)$, which exists by Lemma 6(j). Then $e_1 + e_2 + R_1 + e_4$ is the required hamilton cycle. (See Figure 8(a).)

If $r(y_i, 1) \geq 3$, then it also contains vertex $u''(y_i, 1) = w(y_i, 1)$, by Lemma 1. Thus, since $G_n[y_i, 1, 1]$ is complete, it contains a hamilton path H' from $u(y_i, 1)$ to $w(y_i, 1)$ containing e_1. Let e_5 and e_6 be the edges

$$e_5 = w(y_i, 1)u(y_{i+2}, 1),$$
$$e_6 = w(y_i, 1)v(y_{i+1}, 1),$$

which exist by Lemma 6(k, i). Since $G_n[x, 2, i+1]$ contains either R_1 or R_2, then one of the following is a hamilton cycle in $G_n[x, 2, i]$ containing e_1 and e_2: $H' + e_2 + R_1 + e_5$ or $H' + e_2 + R_2 + e_6$. (See Figure 8(b,c).)

Figure 7: Special cases when $d = 2$ in Theorem 2.

Case $d = 3$:

In this case, $G_n[x, 3, i]$ cannot fall under any case of Exception A of the CLAIM. If $G_n[x, 3, i]$ is Exception B, then $i + 1 = r(x, 3)$ and $G_n[x, 3, i + 1]$ is trivial. We must find $P(x, 3, i)$ and $Q(x, 3, i)$ paths in $G_n[x, 3, i]$. But these paths exist by Lemma 8 since, by induction, $G_n[y_i, 2, 1]$ satisfies conditions (A) and (B) of that lemma.

Otherwise, $G_n[x, 3, i]$ is the general case of the CLAIM and since $d = 3$ is odd, we must find an $O(x, 3, i)$ cycle containing edges $e_1 = u(y_i, 2)v(y_i, 2)$ and $e_2 = u(y_i, 2)v(y_{i+1}, 2)$. Since $G_n[x, 3, i + 1]$ is nontrivial, by induction, it satisfies (A) or (B) of Lemma 7.

If $G_n[y_i, 2, 1]$ is in the general case of the CLAIM, by induction it has an $E(y_i, 2, 1)$ cycle and therefore by Lemma 7, $G_n[x, 3, i]$ has an $O(x, 3, i)$ cycle. Otherwise, $G_n[y_i, 2, 1]$ is not the general case of the CLAIM. But $G_n[x, 3, i+1]$ is nontrivial, so by Corollary 1(b) it contains $v(y_{i+1}, 2)$. But then by Lemma 2(iv), $v'(y_i, 2) \in V(y_i, 2, 1)$. Clearly, then, $G_n[y_i, 2, 1]$ cannot be Exception B or cases (i) or (ii) of Exception A. So, suppose $G_n[y_i, 2, 1]$ is Exception A(iii). Then $G_n[y_i, 2, 1]$ contains only the vertices $\{u(y_i, 2),\ v(y_i, 2),\ u'(y_i, 2),\ v'(y_i, 2)\}$. Since $u''(y_i, 2) \notin V(y_i, 2, 1)$, by Lemma 2(v), $v(y_{i+2}, 2) \notin L(n)$. Thus either $r = i + 2$ and $G_n[x, 3, i+1]$ is Exception B or $r = i+1$ and therefore by Lemma 4, $G_n[x, 3, i+1]$ has vertex

Figure 8: The $d = 2$ case of Theorem 2.

Figure 9: The $d = 3$ case of Theorem 2.

set either $\{u(y_{i+1}, 2),\ v(y_{i+1}, 2)\}$ or $\{u(y_{i+1}, 2),\ v(y_{i+1}, 2),\ u'(y_{i+1}, 2)\}$. ($G_n[x, 3, i+1]$ trivial was considered earlier.) If $G_n[x, 3, i+1]$ is Exception B, by induction, it has a hamilton path R from $v(y_{i+1}, 2)$ to $u(y_{i+2}, 2)$. See Figure 9(a) for an $O(x, 3, i)$ cycle in this case. If $V(x, 3, i+1)$ =$\{u(y_{i+1}, 2),\ v(y_{i+1}, 2),\ u'(y_{i+1}, 2)\}$, it can be verified that $v(y_i, 2)$ and $u'(y_{i+1}, 2)$ are M-adjacent and Figure 9(b) shows an $O(x, 3, i)$ cycle in this case. An $O(x, 3, i)$ cycle in the remaining case that $V(x, 3, i+1) = \{u(y_{i+1}, 2),\ v(y_{i+1}, 2)\}$ is shown in Figure 9(c), where edges exists by Lemma 7.

Case $d \geq 4$:

If $G_n[x, d, i]$ is Exception B of the CLAIM, then $i+1 = r(x, d)$ and $G_n[x, d, i+1]$ is trivial. By induction, $G_n[y_{i-1}, d-1, 1]$ satisfies (A) and (B) of Lemma 8, so in this case, by Lemma 8, $G_n[x, d, i]$ has both $P(x, d, i)$ and $Q(x, d, i)$ paths, as required.

Otherwise, $G_n[x, d, i+1]$ is nontrivial, so by Corollary 1(b) it contains $v(y_{i+1}, d-1)$ and therefore by Lemma 2(iv), $v'(y_i, d-1) \in L(n)$. Thus, $G_n[y_i, d-1, 1]$ is in the general case of the CLAIM, so by induction, it has an $E(y_i, d-1, 1)$ cycle if $d-1$ is even and an $O(y_i, d-1, 1)$ cycle if $d-1$ is odd. Furthermore, since $G_n[x, d, i+1]$ is nontrivial, it satisfies (A) or (B) of Lemma 7 and therefore by Lemma 7, $G_n[x, d, i]$ has an $E(x, d, i)$ cycle if d is even and an $O(x, d, i)$ cycle if d is odd. This completes the proof of the theorem. □

An example of the construction of the theorem is shown in Figure 10. The tree of 10-bit necklaces is shown at the top, from levels 0 through 5. In the center is the hamilton cycle C in $G_{10}[0^{10}, 5, 1]$, resulting from the construction of the theorem. At the bottom is the Gray code obtained by replacing each x on C by $M[x]$.

4 The Algorithm

The proof of Theorem 2 gives a recursive procedure for constructing a Gray code for necklaces of fixed density. The procedure has been implemented in C and is included in the appendix to [Wan]. (A subsequent modification requires storage only O(n).) In this section, we show the time required is $O(nN(n, d))$, where $N(n, d)$ is the number of n bit necklaces of density d.

Below, we give a crude outline of the procedure CYCLE(x, d, i) for constructing a hamilton cycle in the graph $G_n[x, d, i]$. For simplicity, we ignore differences between the different types of cycles (O, E) since these do not affect the time analysis.

Figure 10: The hamilton cycle in $G_{10}[0^{10}, 5, 1]$ and the corresponding Gray code for 10-bit necklaces of density 5.

```
CYCLE(x,d,i)

1.    compute y_i, the i-th child of x
2.    if d=1 then complete graph
3.    else if i=r(x,d) then complete graph
      else if d=2 then
4.       if G[x,d,i+1] is trivial then link cycle in complete graph
            G[y_i,d-1,1] with vertex as in Figure 7(a)
5.       else if G[x,d,i] is exception A(iii) then construct cycle
            as in Figure 7(b)
6.       else if r(y_i,d-1)=2 then link vertices to CYCLE(x,d,i+1)
            as in Figure 8(a)
7.       else link CYCLE(y_i,d-1,1) with CYCLE(x,d,i+1) as in
            Figures 8(b,c)
      else if d=3 then
8.       if G[x,d,i] is Exception B then link CYCLE(y_i,d-1,1) with
            vertex as in Lemma 8
9.       else if G[y_i,d-1,1] is Exception A(iii) then
10.         if G[x,d,i+2] is empty then construct cycle
               as in Figure 9(b,c)
11          else link vertices to CYCLE(x,d,i+1) as in Figure 9(a)
12.      else link CYCLE(y_i,d-1,1) and CYCLE(x,d,i+1) as in Lemma 7
      else {d >= 4 }
13.      if G[x,d,i] is Exception B then link CYCLE(y_i,d-1,1) and vertex as
            in Lemma 8
14.      else link CYCLE(y_i,d-1,1) with CYCLE(x,d,i+1) as in Lemma 7
```

Recall that $L(n)$ is the set of lexicographically smallest representatives of the n-bit necklaces. It is shown in [Shi] that for an arbitrary n-bit string, x, it is possible to check whether $x \in L(n)$ in time $O(n)$. Using this fact, we show that all the tests in the CYCLE algorithm can be made in time $O(n)$.

First note that by definition of $r(x,d)$ and Corollary 1(a), $r(x,d) \geq t$ if and only if $u(y_t, d-1) \in L(n)$, so the tests on lines (3) and (6) can be made in time $O(n)$ using the algorithm of [Shi]. By Corollary 1(a), $G_n[x,d,i]$ is empty if $u(y_i, d-1) \notin L(n)$ and is trivial if and only if $u(y_i, d-1) \in L(n)$, but $v(y_i, d-1) \notin L(n)$. By Corollary 1(b) and Lemma 2(iii), $G_n[x,d,i]$ has only two vertices if and only if $v(y_i, d-1) \in L(n)$, but $w(y_i, d-1), u'(y_i, d-1) \notin L(n)$. Thus, tests in lines (4) and (10) take time $O(n)$. Finally, $G_n[x,d,i]$ is Exception B if and only if $i = r(x,d)-1$ and $G_n[x,d,i+1]$ is trivial; $G_n[x,d,i]$ is Exception A(iii) if and only if $d=2$, $r(y_i, d-1) = 2$, and $r(y_{i+1}, d-1) = 2$. Thus, tests on lines (5), (8), (9), and (13) can be done in time $O(n)$ by testing whether certain binary strings are in $L(n)$. This means that the CYCLE procedure spends no more than $O(n)$ time to determine whether to make a recursive call.

If CYCLE(x,d,i) makes no recursive call, it takes one of the branches (2), (3), (4), (5), or (10), each of which can be implemented in time $O(n|V(x,d,i)|)$.

If CYCLE(x,d,i) makes a recursive call, it does so to one or both of the disjoint subgraphs $G_n[y_i, d-1, 1]$ (a left call), $G_n[x,d,i+1]$ (a right call), and never to a trivial graph. To count the number of recursive subcalls over the entire execution, consider the subtree, $RICH(x,d,i)$ of $TREE(n)$ consisting of all nodes x with descendants in $V(x,d,i)$. Note that no recursive call is made on any node not in $RICH(x,d,i)$. Further, if w is a nonleaf node with only one child in $RICH(x,d)$, no recursive call is made at w (line (2) or (3)). Thus, the number of recursive calls is at most the number of nodes in $RICH(x,d,i)$ with at least two children and this number cannot exceed the number of leaves of $RICH(x,d,i)$, which is $|V(x,d,i)|$.

In summary, the total time for CYCLE(x,d,i) is

$$O(n * \text{ number of recursive calls } + \sum n * |V(z,d',j)|)$$

where the sum on the right is over every call CYCLE(z,d',j) which does not itself make a recursive call. Both terms are $O(n|V(x,d,i)|)$.

In particular, for $d \geq 2$, CYCLE$(x, d-1, 1)$ with $x = 0^{n-1}1$, gives a Gray code for $V(0^{n-1}, d-1, 1) = L(n,d)$ in time $O(n * N(n,d))$.

We mention that to avoid storing cycles, when recursively constructed "left" and "right" cycles are to be linked by two edges, the procedure recursively computes and outputs the left cycle (in an appropriate order), then one link edge is output, and then the right cycle is computed recursively and output (in an appropriate order). To complete the cycle, the second link edge is output. Thus the additional storage required is no more than the depth of the recursion which is $O(n)$.

Although the time analysis of the algorithm can be made tighter in several places, we have found no way to reduce the overall time bound of $O(n * N(n,d))$, either by a tighter analysis or by an alternative implementation. Even for the simpler problem of listing n-bit necklaces of fixed density d in *any* order, no asymptotically faster algorithm is yet known.

Acknowledgement We are grateful to the anonymous referee who offered many suggestions to improve the presentation.

References

[BuWi] M. Buck and D. Wiedemann, "Gray codes with restricted density," *Discrete Mathematics* 48 (1984) 163-171.

[ChChCh] C. C. Chang, H. Y. Chen and C. Y. Chen, "Symbolic Gray code as a data allocation scheme for two-disc systems," *The Computer Journal* 35, 3 (1992) 299-305.

[ChLeDu] C. C. Chang, R. C. T. Lee, and M. W. Du, "Symbolic Gray code as a perfect multiattribute hashing scheme for partial match queries," *IEEE Transactions on Software Engineering* SE - 8, 3 (May 1982) 235-249.

[FrKe] H. Fredricksen and I. J. Kessler, "An algorithm for generating necklaces of beads in two colors," *Discrete Mathematics* 61 (1986) 181-188.

[FrMa] H. Fredricksen and J. Maiorana, "Necklaces of beads in k colors and k-ary de Bruijn sequences," *Discrete Mathematics* 23, 3 (1978) 207-210.

[Gar] M. Gardner, "The curious properties of the Gray code and how it can be used to solve puzzles," *Scientific American* 227, 2 (August 1972) 106-109.

[Gra] F. Gray, "Pulse code communication," U. S. Patent 2632058 (March 17, 1953).

[Joh] S. M. Johnson, "Generation of permutations by adjacent transpositions," *Mathematics of Computation* 17 (1963), 282-285.

[Kay] R. Kaye, "A Gray code for set partitions," *Information Processing Letters* 5 (1976) 171-173.

[Kli] P. Klingsberg, "A Gray code for compositions," *Journal of Algorithms* 3, (1982) 41-44.

[Los] R. M. Losee, "A Gray code based ordering for documents on shelves," *Journal of American Society for Information Science* 43, 4 (1992) 312-322.

[Luc] J. M. Lucas, "The rotation graph of binary trees is hamiltonian," *Journal of Algorithms* 8 (1987) 503-535.

[LuRoRu] J. M. Lucas, D. Roelants van Baronaigien and F. Ruskey, "On rotations and the generation of binary trees," *Journal of Algorithms* 15, 3 (November 1993) 343-366.

[NiWi] A. Nijenhuis and H. S. Wilf, *Combinatorial Algorithms for Computers and Calculators*, Academic Press Inc., New York, (1978).

[PrRu1] G. Pruesse and F. Ruskey, "Generating the linear extensions of certain posets by transpositions," *SIAM Journal on Discrete Mathematics* 4, 3 (August 1991) 413-422.

[PrRu2] G. Pruesse and F. Ruskey, "Generating linear extensions fast," *SIAM Journal on Computing*, to appear.

[RaSaWe] D. Rasmussen, C. Savage, and D. West, "Gray Codes for Families of Integer Partitions," to appear in *Journal of Combinatorial Theory, Series A*.

[Ric] D. Richards, "Data compression and Gray-code sorting," *Information Processing Letters* 22 (1986) 201-205.

[Rus1] F. Ruskey, "Adjacent interchange generation of combinations," *Journal of Algorithms* 9, 2 (June 1988) 162-180.

[Rus2] F. Ruskey, "Generating linear extensions of posets by transpositions," *Journal of Combinatorial Theory (B)* 54, 1 (1992) 77-101.

[RuPr] F. Ruskey and A. Proskurowski, "Generating binary trees by transpositions," *Journal of Algorithms* 11, 1 (1990) 68-84.

[RuSaWa] F. Ruskey, C. D. Savage, and T. M. Wang, "Generating necklaces," *Journal of Algorithms* 13, 3 (September 1992) 414-430.

[Sav] C. D. Savage, "Gray code sequences of partitions," *Journal of Algorithms* 10, 4 (1989) 577-595.

[Shi] Y. Shiloach "Fast canonization of circular strings" *Journal of Algorithms* 2, 2 (1981) 107-121.

[Sta] G. Stachowiak, "Hamilton paths in graphs of linear extensions for unions of posets," *SIAM J. Discrete Math.* 5 (1992), 199-206.

[Tro] H. F. Trotter, "PERM (Algorithm 115)", *Communications of the ACM* 5, 8 (1962) 434-435.

[Wes] D. B. West, "Generating linear extensions by adjacent transpositions," *Journal of Combinatorial Theory (B)*, 57(1993).

[WaSa] T. M. Wang and C. D. Savage, "A new algorithm for generating necklaces," *Proceedings, Twenty-eighth Annual Allerton Conference on Communication, Control, and Computing*, Allerton, October 1990.

[Wan] T. M. Wang, "Gray codes for necklaces of fixed density," Ph.D. Thesis, Department of Computer Science, North Carolina State University (1994).

[Wil] H. S. Wilf, *Combinatorial Algorithms: An Update*, SIAM, Philadelphia (1989).