Generating Planar Paths in Convex Position in Constant Amortized Time*

Ro–Yu Wu[1] Jou–Ming Chang[2,†] Kung–Jui Pai[3,4] and Yue–Li Wang[3]

[1] Department of Industrial Management,
Lunghwa University of Science and Technology, Taoyuan, Taiwan, ROC
[2] Department of Information Management,
National Taipei College of Business, Taipei, Taiwan, ROC
[3] Department of Information Management,
National Taiwan University of Science and Technology, Taipei, Taiwan, ROC
[4] Department of Industrial Engineering and Management,
Mingchi University of Technology, Taipei, Taiwan, ROC

Abstract

Let S be a set of $n \geqslant 3$ points arranged in convex position in the plane and suppose that all points of S are labeled from 1 to n in clockwise direction. A planar path P on S is a path whose edges connect all points of S with straight line segments and such that no two edges of P cross. Flipping an edge on P means that a new path P' is obtained from P by a single edge replacement. The geometric path graph on S is a graph consisting of all planar paths as its vertices and two vertices are joined by an edge if their corresponding paths can be transformed to each other by using a flip. In this paper, we provide efficient algorithms to generate all planar paths. With the help of the loopless generation of a set of 2-way binary-reflected Gray codes, the proposed algorithms generate the next planar path by means of a flip and such that the number of position changes for points in the path has a constant amortized upper bound. Furthermore, as a by-product, we show the existence of a Hamiltonian cycle in geometric path graphs.

Keywords: Combinatorial problems; Enumerating algorithms; Convex position; Flips; Gray codes; Planar paths; Amortized analysis

1. Introduction

Let S be a set of $n \geqslant 3$ points in convex position in the plane and suppose that all points of S are labeled from 1 to n in clockwise direction. A *planar path* P is a Hamiltonian path (i.e., a spanning path of S) whose edges connect all points of S with straight line segments and such that no two edges of P cross. We denote a planar path passing through points $p_0, p_1, \ldots, p_{n-1}$ by $p_0p_1 \cdots p_{n-1}$ and regard the paths $p_0p_1 \cdots p_{n-1}$ and $p_{n-1}p_{n-2} \cdots p_0$ to be equivalent for the undirected case. Let $\mathscr{P}(S)$ denote the set of all planar paths of S. Flipping an edge on a path $P \in \mathscr{P}(S)$ means that a new path $P' \in \mathscr{P}(S)$ is obtained from P by a single edge replacement. In this way, we say that P' is obtained from P by means of a *flip*. The *geometric path graph* on S, denoted by $G(S)$, is a graph consisting of all paths in $\mathscr{P}(S)$ as its vertices and two vertices are joined by an edge if their corresponding paths can be transformed to each other by

*This research was partially supported by National Science Council under the Grants NSC97-2115-M-141-001-MY2.

†Corresponding author. (spade@mail.ntcb.edu.tw)

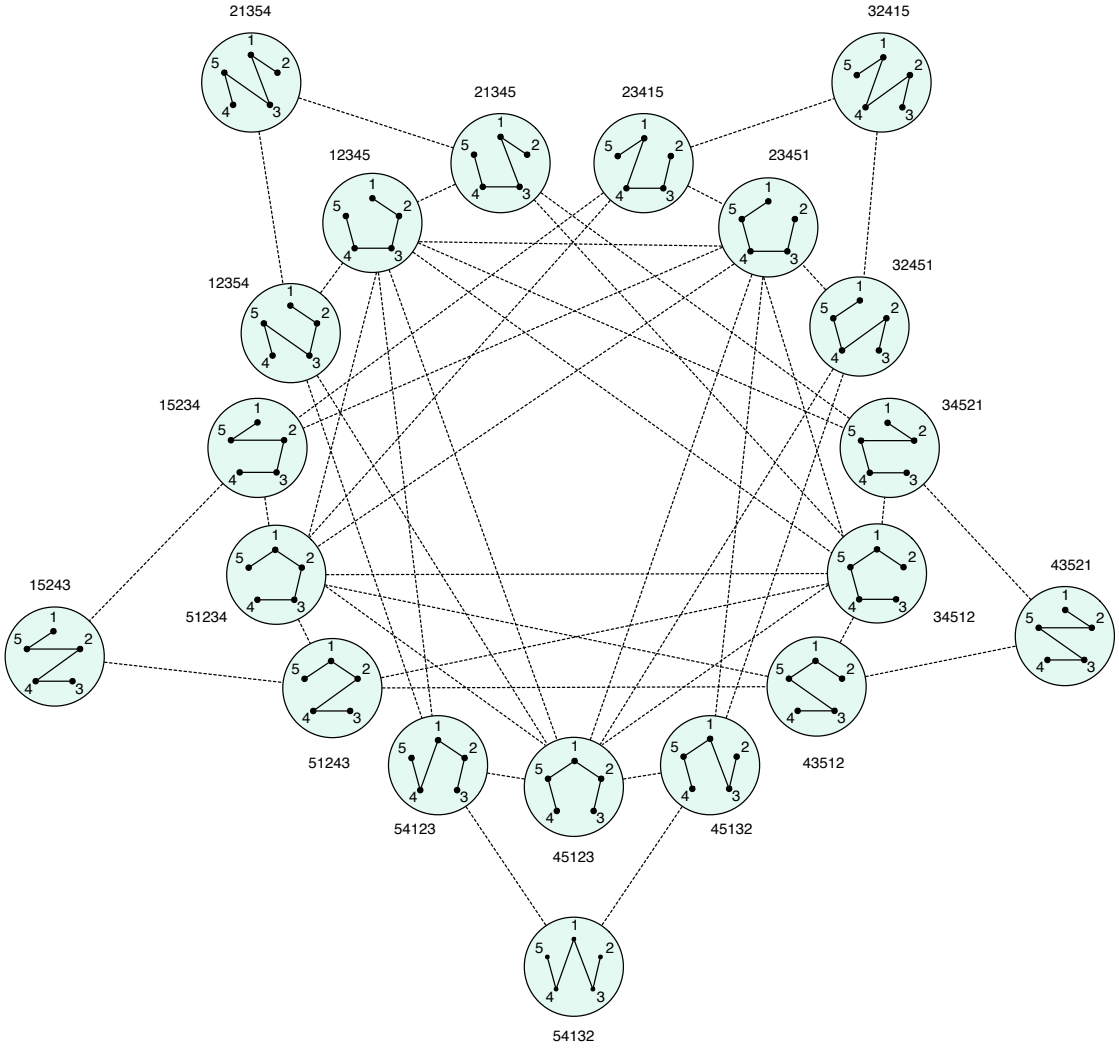

Figure 1: The geometric path graph G_5.

using a flip. Note that $G(S)$ is isomorphic to G_n, where G_n is the geometric path graph on the set of points of a regular n-gon. For example, Figure 1 shows the geometric path graph G_5.

Akl et al. [3] recently studied several problems related to the planar paths in convex position. They pointed out that the number of planar paths in $\mathscr{P}(S)$ is totally $n2^{n-3}$ and every planar path can be transformed into another planar path by means of at most $2n-5$ flips. This implies that G_n is connected and has diameter no more than $2n-5$. Later on, Chang and Wu [9] searched out the exact diameter of G_n for $n \geqslant 3$. In addition, experimental results were also given in [3] for discussing the connectedness of geometric path graphs of points in general position. For more results concerning with planar paths and their generalization, called *planar trees*, on a set of points in convex position and in general position, the reader can refer to [1, 2, 4, 5, 8, 13]

In this paper, we will be interested on the problem of generating planar paths in convex position. Avis and Fukuda [5] proposed an algorithm for enumerating planar trees in

general position by a so-called *reverse search* technique. The aim of their algorithm is to pursue the one with a simple description instead of searching for the most efficient one. Consequently, the time required between two successive planar trees is bounded in $\mathcal{O}(n^3)$. Using a certain 0/1 encoding to represent the direction of a planar path along with the initial point p_0, Akl et al. [3] also proposed an algorithm employing flips and 2-flips (i.e., an operation for path transformation using two edges replacement) to generate all directed planar paths on a set of n points in convex position. They further showed that it is possible to generate each directed planar path in $\mathcal{O}(n)$ time. Here we focus on the problem of generating undirected planar paths. In particular, we use naive integer sequences of length n to encode such planar paths.

Our algorithm fleetly produces the next planar path by means of a flip and mainly relies on the following two specific functions. One is a loopless function for generating a particular binary reflected Gray code in a constant time for each generation, and the other is a transformation that coverts a Gray code into the desired integer sequence for representing the corresponding planar path. However, caused by the definition of flip itself, the worst case time of the conversion in the latter function is proportional to n (i.e., the length of the encoding of a planar path). From an analysis of aggregate method, we will show that our algorithm has amortized efficiency, where each generation of planar path takes an amortized cost of no more than 7/3 position changes for n odd and no more than 19/8 position changes for n even, respectively. Furthermore, as a by-product, we obtain an interesting consequence competitive with the result in [17] that showed the existence of hamiltonicity in geometric path graph G_n for $n \geqslant 3$.

2. Loopless generation of 2-way binary-reflected Gray codes

A *binary Gray code* of length m is a list $s(0), s(1), \ldots, s(2^m-1)$ of the 2^m distinct m-bit strings of 0s and 1s, with the property that each $s(i)$ differs from $s(i+1)$ in only one bit. A binary Gray code is said to be *cyclic* if $s(2^m-1)$ and $s(0)$ in the list also differ by only one bit. The *binary-reflected Gray code* (BRGC for short) is a particular binary Gray code that can be generated recursively by reflecting the bits (i.e., listing them in reverse order and concatenating the reverse list onto the original list), prefixing the original bits with a binary 0 and then prefixing the reflected bits with a binary 1. The base case is the list $s(0) = 0$, $s(1) = 1$ for $m = 1$. BRGC was first designed to speed up telegraphy by Frank Gray [12], but now has numerous applications including databases, experimental design, and puzzle solving [10, 11, 15, 16, 19]. Especially, it was referred more generally to an exhaustive listing of combinatorial objects. An excellent survey for listing combinatorial objects in Gray code order is that of Savage's [18].

We now introduce a variation of BRGC as follows. Let $s(i) = b_{m-1}b_{m-2}\cdots b_0$ be the $(i+1)$th string in the list of BRGC. A *2-way binary-reflected Gray code* (abbreviated as 2BRGC) of length m is a list $s'(0), s'(1), \ldots, s'(2^m-1)$ of the 2^m distinct m-bit strings of 0s and 1s, where $s'(i)$ is defined by

$$s'(i) = \begin{cases} b_1 b_3 \cdots b_{m-2} b_{m-1} b_{m-3} \cdots b_2 b_0 & \\ & \text{if } m \text{ is odd;} \\ b_1 b_3 \cdots b_{m-3} b_{m-1} b_{m-2} \cdots b_2 b_0 & \\ & \text{if } m \text{ is even.} \end{cases}$$

That is, $s'(i)$ is obtained from $s(i)$ by rearranging m bits such that all bits with odd position are located in the front of the string by increasing their indices and all bits with even position are located in the rear of the string

by decreasing their indices. Note that a fairly involved web search has turned up no results on a similar definition to 2BRGC. Table 1 will help to illustrate the construction of 2BRGC of any length m.

Table 1: The BRGC and 2BRGC of length 4.

i	$s(i)$	$s'(i)$
0	0000	0000
1	0001	0001
2	0011	1001
3	0010	1000
4	0110	1010
5	0111	1011
6	0101	0011
7	0100	0010
8	1100	0110
9	1101	0111
10	1111	1111
11	1110	1110
12	1010	1100
13	1011	1101
14	1001	0101
15	1000	0100

A *loopless algorithm* for generating combinatorial objects is allowed to have loops in its initialization step, as long as the initial setup is reasonably efficient; after all, every generation takes a constant time (i.e., only assignment statements and "if-then-else" statements are allowed). The earliest loopless algorithm for listing BRGC is due to Bitner et al. [7] (see also [14]). In what follows, we give a loopless generation of 2BRGC, where our design scheme is inspired from the framework of Knuth [14, Algorithm L].

Algorithm GEN-2BRGC. // This algorithm visits all binary m-bit strings $b_{m-1}b_{m-2}\cdots b_0$ in the order of the 2-way BRGC. It uses an array of "focus pointers" $(f_{m-1}, f_{m-2}, \ldots, f_0)$ to keep track of positions for bits in the string to be complemented.

begin

1. [Initialize.]
   Set $b_j \leftarrow 0$ and $f_j \leftarrow j$ for $0 \leqslant j < m$.
   Set $f_m \leftarrow m$.
2. [Visit.]
   Visit the m-bit string $b_{m-1}b_{m-2}\cdots b_0$.
3. [Set focus pointer.]
   Set $j \leftarrow f_0$ and $f_0 \leftarrow 0$.
   Set $k \leftarrow \begin{cases} j/2 & \text{if } j \text{ is even;} \\ m-1-j/2 & \text{otherwise.} \end{cases}$
   Terminate if $j = m$; otherwise set $f_j \leftarrow f_{j+1}$ and $f_{j+1} \leftarrow j+1$.
4. [Complement bit.]
   Set $b_k \leftarrow 1 - b_k$ and goto Step 2.

end GEN-2BRGC

For example, Table 2 shows the computation of 2BRGC when $m = 4$. Obviously, the algorithm needs $2n + \mathcal{O}(1)$ space and uses only two comparisons together with six assignments without regarding to the initialization. Since the only modification in the algorithm is the rearrangement of bit positions, its correctness directly follows from the pioneers [7, 14].

Table 2: Loopless generation of 2BRGC of length 4.

b_3	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0
b_2	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
b_1	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
b_0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0
f_3	3	3	3	3	3	3	3	3	4	4	4	4	3	3	3	3
f_2	2	2	2	2	3	3	2	2	2	2	2	2	4	4	2	2
f_1	1	1	2	1	1	1	3	1	1	1	2	1	1	1	4	1
f_0	0	1	0	2	0	1	0	3	0	1	0	2	0	1	0	4

Lemma 1 *Algorithm* GEN-2BRGC *is a loopless algorithm for producing the list of a 2BRGC of length m.*

3. Planar paths and their laconic labelings

In this section, we will describe a simple binary labeling for planar paths. Then, putting these labelings into an order of 2BRGC, we can easily generate their corresponding planar paths. For a set S of n points in convex position, we denote $ch(S)$ as the boundary of the convex hull of S. Let $P = p_0p_1\cdots p_{n-1}$ be a planar path in $\mathscr{P}(S)$. An edge $p_ip_{i+1} \in P$ is said to be an *arc* if it lies on $ch(S)$, and a *chord* otherwise. An observation has been pointed out in [3] and [17] that p_0p_1 and $p_{n-2}p_{n-1}$ must always be arcs. For this reason, we call p_0p_1 and $p_{n-2}p_{n-1}$ the *terminal arcs* of P.

For each planar path $P \in \mathscr{P}(S)$, it is associated with a labeling $\ell(P) = \ell_1\ell_2\cdots\ell_{n-3}$ where each ℓ_i, $1 \leqslant i \leqslant n-3$, is defined as follows:

$$\ell_i = \begin{cases} 0 & \text{if } p_ip_{i+1} \in P \text{ is an arc;} \\ 1 & \text{if } p_ip_{i+1} \in P \text{ is a chord.} \end{cases}$$

Such a labeling is called the *laconic labeling* of P. In particular, a planar path $P \in \mathscr{P}(S)$ with $\ell(P) = 00\cdots0$ is called a *canonical path*. Note that it is possible to have the same laconic labeling for different planar paths. That is why it is called a labeling, but not an encoding. Two planar paths $P, Q \in \mathscr{P}(S)$ are said to be *rotationally isomorphic*, denoted by $P \cong Q$, if they look the same when one is turned a certain amount of rotations in the convex position without concerning about the labels of points. For example, for $n = 5$, it is easy to check in Figure 1 that the following planar paths with the same laconic labeling are pairwise rotationally isomorphic:

$$\ell(12345) = \ell(34512) = \ell(51234) = \ell(23451) = \ell(45123) = 00;$$

$$\ell(12354) = \ell(34521) = \ell(51243) = \ell(23415) = \ell(45132) = 01;$$

$$\ell(21354) = \ell(43521) = \ell(15243) = \ell(32415) = \ell(54132) = 11;$$

$$\ell(21345) = \ell(43512) = \ell(15234) = \ell(32451) = \ell(54123) = 10.$$

Furthermore, if a terminal arc of a path is designated, we can determine the path by its laconic labeling immediately. As we have mentioned earlier, Akl et al. [3] also provided a set of encodings to represent directed planar paths. Although laconic labeling looks similar to that encoding, these definition are actually quite distinct.

Because our algorithm allows only one flip to be applied in each generation of planar path, we now give the detail about flips in a planar path. For $P, P' \in \mathscr{P}(S)$, we write $P \overset{e'}{\underset{e}{\longrightarrow}} P'$ to denote the flip that transforms P into P' by replacing the edge e with e'. Suppose that $P = p_0p_1\cdots p_{n-1}$ and $e = p_ip_{i+1}$ for $0 \leqslant i \leqslant n-2$. Obviously, if e is a terminal arc, then P must be a canonical path. In this case, either $P' = p_1p_2\cdots p_{n-1}p_0$ or $P' = p_{n-1}p_0p_1\cdots p_{n-2}$. Otherwise, e' and P' might be one of the following:

$$P' = \begin{cases} p_ip_{i-1}\cdots p_0p_{i+1}p_{i+2}\cdots p_{n-1} & \\ \qquad\qquad \text{if } e' = p_0p_{i+1}; & \\ p_0p_1\cdots p_ip_{n-1}p_{n-2}\cdots p_{i+1} & \\ \qquad\qquad \text{if } e' = p_ip_{n-1}; & \\ p_{i+1}p_{i+2}\cdots p_{n-1}p_0p_1\cdots p_i & \\ \qquad\qquad \text{if } e' = p_0p_{n-1}. & \end{cases}$$

Accordingly, the flip in the first two cases can be considered to be a reversal of a subpath of P. For convenience, the former is called the *left-side reversal* with length $i+1$ and is denoted by $P' = \text{REV}^-(P, i+1)$, and the latter is called the *right-side reversal* with length $n-i-1$ and is denoted by $P' = \text{REV}^+(P, n-i-1)$. Except for the above two cases, the flip in the remaining cases can be considered to be a rotation of a subpath of P. Here we do not provide any notation to denote such type of flips because it will be unavailable in our generating algorithm. For instance, Ta-

ble 3 shows a sequence of flips applying to a canonical path 1234567. From the table, it is easy to check that all corresponding laconic labelings of the newly generated planar paths form a cyclic 2BRGC.

Table 3: A sequence of flips apply to a canonical path 1234567.

$\ell(P)$	$P \xrightarrow[e]{e'} P' = \text{REV}^{\pm}(P,k)$
0000	$1234567 \xrightarrow[56]{57} 1234576 = \text{REV}^{+}(1234567, 2)$
0001	$1234576 \xrightarrow[23]{13} 2134576 = \text{REV}^{-}(1234576, 2)$
1001	$2134576 \xrightarrow[57]{56} 2134567 = \text{REV}^{+}(2134576, 2)$
1000	$2134567 \xrightarrow[45]{47} 2134765 = \text{REV}^{+}(2134567, 3)$
1010	$2134765 \xrightarrow[76]{75} 2134756 = \text{REV}^{+}(2134765, 2)$
1011	$2134756 \xrightarrow[13]{23} 1234756 = \text{REV}^{-}(2134756, 2)$
0011	$1234756 \xrightarrow[75]{76} 1234765 = \text{REV}^{+}(1234756, 2)$
0010	$1234765 \xrightarrow[34]{14} 3214765 = \text{REV}^{-}(1234765, 3)$
0110	$3214765 \xrightarrow[76]{75} 3214756 = \text{REV}^{+}(3214765, 2)$
0111	$3214756 \xrightarrow[21]{31} 2314756 = \text{REV}^{-}(3214756, 2)$
1111	$2314756 \xrightarrow[75]{76} 2314765 = \text{REV}^{+}(2314756, 2)$
1110	$2314765 \xrightarrow[47]{45} 2314567 = \text{REV}^{+}(2314765, 3)$
1100	$2314567 \xrightarrow[56]{57} 2314576 = \text{REV}^{+}(2314567, 2)$
1101	$2314576 \xrightarrow[31]{21} 3214576 = \text{REV}^{-}(2314576, 2)$
0101	$3214576 \xrightarrow[57]{56} 3214567 = \text{REV}^{+}(3214576, 2)$
0100	$3214567 \xrightarrow[14]{17} 3217654 = \text{REV}^{+}(3214567, 4)$

To obtain a subset of planar paths in $\mathscr{P}(S)$ in which their corresponding laconic labelings constitute a set of cyclic 2BRGC, we perform the following procedure.

Procedure GROUP-LIST ($P \in \mathscr{P}(S)$: a canonical path).
begin
1. Set $m \leftarrow n-3$, $Q \leftarrow P$ and output P.
2. Call GEN-2BRGC to produce a list $s(0), s(1), \ldots, s(2^m-1)$ of distinct m-bit 0/1 strings and for each $1 \leqslant i \leqslant 2^m - 1$ do the following:
2.1. Let k be the position where bit '1' occurs in $s(i-1) \oplus s(i)$.
2.2. If $2k < m$, then
Set $Q' \leftarrow \text{REV}^{+}(Q, k+2)$;
else
Set $Q' \leftarrow \text{REV}^{-}(Q, m+1-k)$.
2.3. Output Q'.
2.4. Set $Q \leftarrow Q'$.
end GROUP-LIST

In the above procedure, $\oplus$ denotes the exclusive disjunction. In fact, we do not need to compute the value of k since it has been derived in Step 2 of GEN-2BRGC. Recall that the most significant bit and the least significant bit in binary string $s(i)$ have indices $m-1$ and 0, respectively. Thus, $2k < m$ means that the complemented bit b_k is in the right-half of the string. In this case, the next planar path can be generated by reversing the rightmost $k+2$ positions. On the other hand, if the complemented bit is in the left-half of the string, then the next planar path can be generated by reversing the leftmost $m+1-k$ positions. Therefore, we have the following lemma provided that the planar path P is canonical.

Lemma 2 *Procedure* GROUP-LIST *correctly generates a set of planar paths such that their corresponding laconic labelings form a cyclic 2BRGC.*

For a canonical path P, we denote $GL(P)$ as the set of planar paths generated by the procedure GROUP-LIST(P). Then, we can easily prove the following lemma by using the notion of rotational isomorphism.

Lemma 3 *If $P, Q \in \mathscr{P}(S)$ are two distinct canonical paths, then $GL(P) \cap GL(Q) = \emptyset$.*

4. Generating planar paths and amortized analysis

We are now at a position to describe our main result. Recall the previous mention that $p_0p_1\cdots p_{n-1}$ and $p_{n-1}p_{n-2}\cdots p_0$ are regarded as the same planar path. Thus, the change between $p_0p_1\cdots p_{n-1}$ and $p_{n-1}p_{n-2}\cdots p_0$ in our generating algorithm (if necessary) does not take any flip. In particular, if a current generated planar path P is represented by $p_0p_1\cdots p_{n-1}$, we may alternately take the place of $p_{n-1}p_{n-2}\cdots p_0$ and admit the exchange of directions for the left-side and the right-side in the succeeding generation, such as the call for functions REV$^+$() and REV$^-$(). The following two algorithms are used to generate all planar paths when n is odd and even, respectively.

Algorithm GEN-PLANAR-PATHS-ODD.
// This algorithm generates all planar paths in $\mathscr{P}(S)$ when n is odd.
begin
1. Set $P \leftarrow 12\cdots n$.
2. For each $i = 1, 2, \ldots, n$ do the following
2.1. Call GROUP-LIST(P) and let Q be the last generated planar path.
2.2. Set $P \leftarrow$REV$^+(Q, \lceil n/2 \rceil)$.

end GEN-PLANAR-PATHS-ODD

Algorithm GEN-PLANAR-PATHS-EVEN.
// This algorithm generates all planar paths in $\mathscr{P}(S)$ when n is even.
begin
1. Set $P \leftarrow 12\cdots n$.
2. For each $i = 1, 2, \ldots, n/2$ do the following
2.1. Call GROUP-LIST*(P) and let Q be the last generated planar path.
2.2. Set $P \leftarrow$REV$^-(Q, n/2)$.
2.3. Call GROUP-LIST(P) and let Q be the last generated planar path.
2.4. Set $P \leftarrow$REV$^-(Q, n/2)$.
2.5. Output P.
2.6. Set $P \leftarrow$REV$^+(P, n-2)$.

end GEN-PLANAR-PATHS-ODD

In the second algorithm, the procedure GROUP-LIST* in Step 2.1 is similar to that of GROUP-LIST except for lacking the output of the initial planar path P. For example, Figures 2 and 3 show all generated planar paths in convex position for $n = 7$ and $n = 6$, respectively.

Theorem 1 *Algorithms* GEN-PLANAR-PATHS-ODD *and* GEN-PLANAR-PATHS-EVEN *will produce all planar paths of n points in convex position for n odd and for n even, respectively. In particular, every generated planar path can be achieved by a flip that has an amortized cost with at most $\frac{7}{3}$ position changes for n odd and at most $\frac{19}{8}$ position changes for n even, respectively.*

Proof. First, an easy observation shows that $\mathscr{P}(S)$ contains n canonical paths. By Lemmas 2 and 3, we can partition the total $n2^{n-3}$ planar paths of $\mathscr{P}(S)$ into n groups and such that all planar paths in each group have distinct laconic labelings. Since Step 2.2 in GEN-PLANAR-PATHS-ODD and Steps 2.2 and 2.4 in GEN-PLANAR-PATHS-EVEN can generate a next canonical path after invoking a call of GROUP-LIST, this shows how the sequence of inter-group listing is carried out. Thus, it is clear that every planar path in $\mathscr{P}(S)$ is generated exactly once by means of a flip.

To analyze the algorithms, we perform an amortized analysis to compute the number of position changes in a flip. We first suppose that n is odd and consider the list $s(0), s(1), \ldots, s(2^m - 1)$ of the 2BRGC of length $m = n - 3$. For convenience, we let

$$s(i) = b_1^i b_3^i \cdots b_{m-3}^i b_{m-1}^i b_{m-2}^i \cdots b_2^i b_0^i.$$

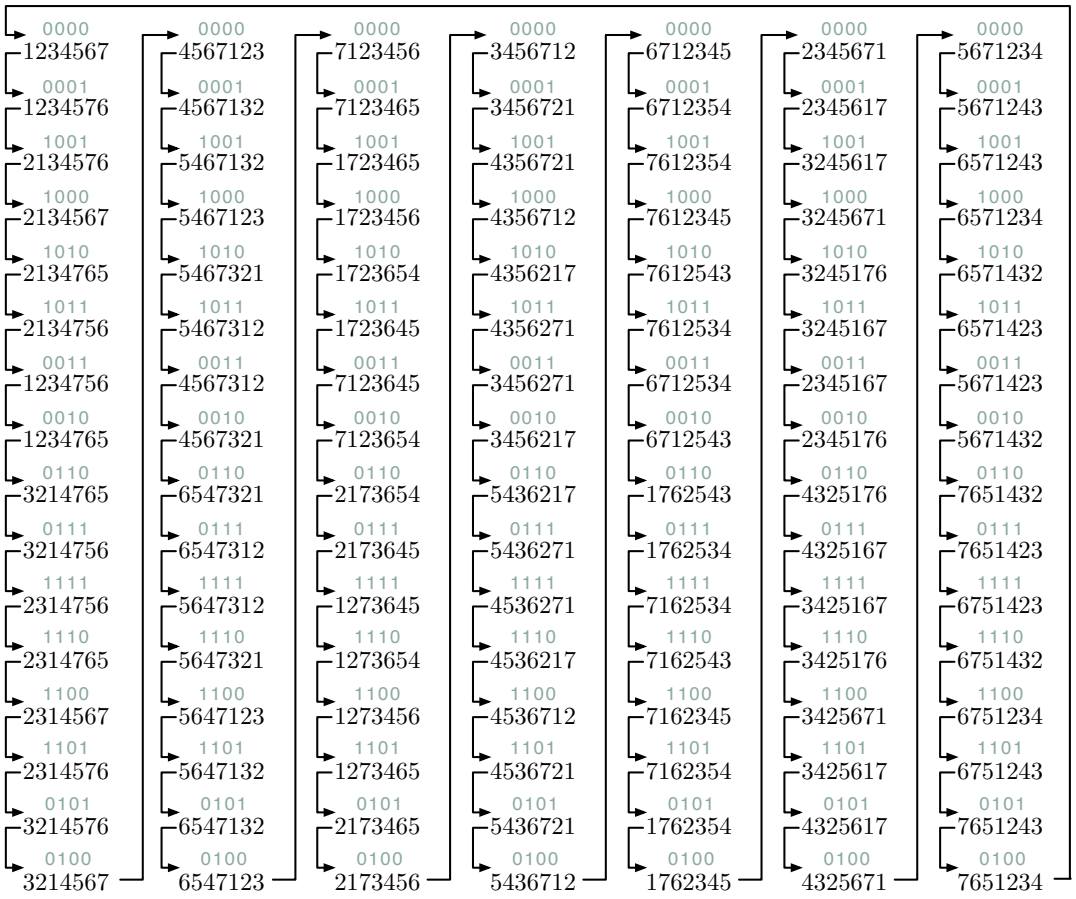

Figure 2: All generated planar paths of 7 points in convex position.

and say that b_k^i has position k. For each $k = 0, 1, \ldots, m-1$, the *transition count* of position k is defined by

$$c(k) = \sum_{j=1}^{2^m-1} b_k^{j-1} \oplus b_k^j.$$

That is, $c(k)$ is the number of bit changes on position k. Since the 2BRGC is obtained from the BRGC by arranging bit positions, the transition count is easily derived from the original, which is given by

$$c(k) = 2^{m-(k+1)}$$

for $0 \leqslant k \leqslant m-1$ [6, 19]. As we have known from Procedure Group-List, a change of bit in position k will produce a flip that reverses a subpath of length $\lceil \frac{k+3}{2} \rceil$ in a planar path. Beyond that, an additional flip takes the reversal with length $\lceil n/2 \rceil = m/2 + 2$ between two calls of Group-List. Thus, for every round in Step 2 of Algorithm Gen-Planar-Paths-Odd, the total number of position changes is

$$\begin{aligned} C_{\text{odd}} &= \sum_{k=0}^{m-1} c(k) \cdot \left\lceil \frac{k+3}{2} \right\rceil + \left(\frac{m}{2}+2\right) \\ &= 2(2^{m-1}+2^{m-2}) + 3(2^{m-3}+2^{m-4}) + \cdots \\ &\quad + \left(\frac{m}{2}+1\right)(2^1+2^0) + \left(\frac{m}{2}+2\right) \\ &= 2^m \left(2 \cdot \frac{3}{4} + 3 \cdot \frac{3}{4^2} + \cdots + \left(\frac{m}{2}+1\right) \cdot \frac{3}{4^{\frac{m}{2}}}\right) \\ &\quad + \left(\frac{m}{2}+2\right). \end{aligned} \tag{1}$$

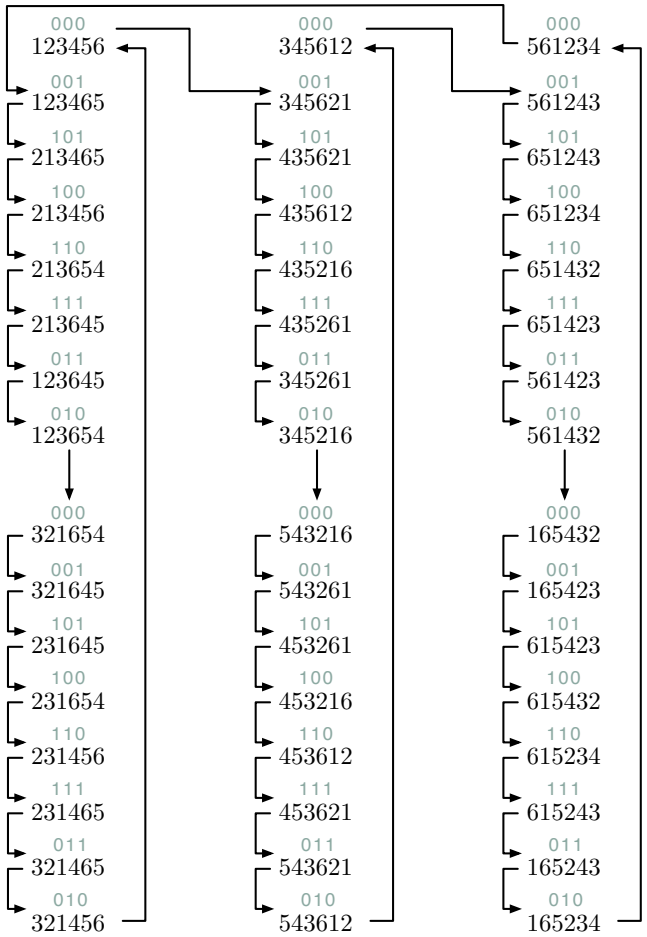

Figure 3: All generated planar paths of 6 points in convex position.

By simplifying the series in the first term of Eq. (1), we obtain an upper bound of the average number of position changes in a flip as follows:

$$\begin{aligned}\frac{C_{\text{odd}}}{2^m} &= \frac{7}{3}-\left(\frac{m}{2}+\frac{7}{3}\right)\left(\frac{1}{2^m}\right)+\left(\frac{m}{2}+2\right)\left(\frac{1}{2^m}\right)\\ &= \frac{7}{3}-\frac{1}{3}\left(\frac{1}{2^m}\right)\\ &< \frac{7}{3}.\end{aligned}$$

Similarly, if n is an even integer, the total number of position changes in Gen-Planar-Paths-Even is

$$\begin{aligned}C_{\text{even}} &= \sum_{k=0}^{m-1} c(k)\cdot\left\lceil\frac{k+3}{2}\right\rceil+\left(\frac{m+3}{2}\right)\\ &\quad+\frac{((m+1)-2)+(2-2)}{2}\\ &= 2(2^{m-1}+2^{m-2})+3(2^{m-3}+2^{m-4})+\cdots\\ &\quad+\left(\frac{m+1}{2}\right)(2^2+2^1)+(m+3)+\frac{m-1}{2}\\ &= 2^m\left(2\cdot\frac{3}{4}+3\cdot\frac{3}{4^2}+\cdots+\left(\frac{m+1}{2}\right)\cdot\frac{3}{4^{\frac{m-1}{2}}}\right)\\ &\quad+\frac{3}{2}m+\frac{5}{2}. \qquad (2)\end{aligned}$$

We can simplify Eq. (2) to obtain a similar result as follows:

$$\begin{aligned}\frac{C_{\text{even}}}{2^m} &= \frac{7}{3}-\left(m+\frac{11}{3}\right)\left(\frac{1}{2^m}\right)+\left(\frac{3}{2}m+\frac{5}{2}\right)\left(\frac{1}{2^m}\right)\\ &= \frac{7}{3}+\left(\frac{m}{2}-\frac{7}{6}\right)\left(\frac{1}{2^m}\right)\\ &\leqslant \frac{19}{8}\end{aligned}$$

where the term $(\frac{m}{2}-\frac{7}{6})(1/2^m)$ has the maximum value $\frac{1}{24}$ when $m=3$ or $m=5$ is an odd integer. □

5. Concluding remarks

In this paper, we use a naive integer sequence to represent a planar path in a convex position. This representation is conceptually simple. With the help of generating laconic labelings of planar paths in the order of a 2BRGC, we can easily generate all planar paths in a systematic way. Consequently, we show that every generated planar path can be achieved by a flip with a constant amortized upper bound of position changes. Moreover, since procedure Group-List produces a cyclic 2BRGC and Algorithms Gen-Planar-Paths-Odd and Gen-Planar-Paths-Even properly switch the group listing of planar paths, this gives an alternative proof of Rivera-Campo and Urrutia-Galicia's result that a Hamiltonian cycle exists in any geometric path graph G_n for $n \geqslant 3$.

References

[1] O. Aichholzer, F. Aurenhammer, F. Hurtado, Sequences of spanning trees and a fixed tree theorem, *Computational Geometry* 21 (2002) 3–20.

[2] O. Aichholzer, K. Reinhardt, A quadratic distance bound on sliding between crossing-free spanning trees, *Computational Geometry* 37 (2007) 155–161.

[3] S.G. Akl, K. Islam, H. Meijer, On planar path transformation, *Information Processing Letters* 104 (2007) 59–64.

[4] S.G. Akl, K. Islam, H. Meijer, Planar tree transformation: results and counterexample, *Information Processing Letters* 109 (2008) 61–67.

[5] D. Avis, K. Fukuda, Reverse search for enumeration, *Discrete Applied Mathematics* 65 (1996) 21–46.

[6] G.S. Bhat, C.D. Savage, Balanced Gray codes, *Electronic Journal of Combinatoric* 3 (1996) #R25.

[7] J.R. Bitner, G. Ehrlich, E.M. Reingold, Efficient generation of the binary reflected Gray code and its applications, *Communication of the ACM* 19 (1976) 517–521.

[8] P. Bose, F. Hurtado, Flips in planar graphs, *Computational Geometry* 42 (2009) 60–80.

[9] J.-M. Chang, R.-Y. Wu, On the diameter of geometric path graphs of points in convex position, *Information Processing Letters* 109 (2009) 409–413.

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

[11] L. Goddyn, G.M. Lawrence, E. Nemeth, Gray codes with optimized run lengths, *Utilitas Mathematica* 34 (1988) 179–192.

[12] F. Gray, Pulse code communication, U.S. Patent, 2632058, 1953.

[13] C. Hernando, F. Hurtado, A. Márquez, M. Mora, M. Noy, Geometric tree graphs of points in convex position, *Discrete Applied Mathematics* 93 (1999) 51–66.

[14] D.E. Knuth, The Art of Computer Programming. Volume 4 Fascicle 2 – Generating All Tuples and Permutations, Addison-Wesley, 2005.

[15] J.E. Ludman, Gray code generation for MPSK signals, *IEEE Transactions on Communications* COM-29 (1981) 1519–1522.

[16] J.E. Ludman, J.L. Sampson, A technique for generating Gray codes, *Journal of Statistical Planning and Inference* 5 (1981) 171–180.

[17] E. Rivera-Campo, V. Urrutia-Galicia, Hamilton cycles in the path graph of a set of points in convex position, *Computational Geometry* 18 (2001) 65–72.

[18] C.D. Savage, A survey of combinatorial Gray codes, *SIAM Review* 39 (1997) 605–629.

[19] V.E. Vickers, J. Silverman, A technique for generating specialized Gray sodes, *IEEE Transactions on Computers* C-29 (1980) 329–331.