"We give a formula for the optimal sequence of perfect shuffles to bring a card at position p to the top (or indeed, to any position). This solves a fifty-year-old problem of Elmsley. The argument illustrates elementary group theory and shows how a simple card trick can lead to the edge of what is known."

The Solutions to Elmsley's Problem

Persi Diaconis and Ron Graham
Stanford University and University of California, San Diego

A handful of magicians and gamblers can shuffle cards perfectly. This means cutting the deck exactly in half and riffling the cards together so that they alternate perfectly. Figure 1 shows a perfect shuffle of a deck of ten cards labeled from top to bottom by 0, 1, 2, ..., 9.

Figure 1. out shuffle of a ten card deck.

Perfect shuffles are of many uses. For example, eight perfect **out** shuffles of a 52 card pack bring the deck back to its original order. In Figure 1, the original top card (labeled 0) stays at the top of the pack. (Hence, for an **out** shuffle, the top card remains **out**side.) There is a second type of perfect shuffle, called an **in** shuffle, where the original top card winds up second from the top (**in**side) as in Figure 2.

Here is an application of **in** shuffles. Imagine the four Aces sitting on top of the deck. After one **in** shuffle, they are every second card. After two **in** shuffles, they are every fourth card. Hence, if the deck is dealt into four hands, one card at a time, the dealer gets the Aces.

For these and other reasons, gamblers and magicians have studied the properties of perfect shuffles for close to 300 years. We develop some properties we will need in the next section.

It is natural to ask what can be done by combining **in** and **out** shuffles. For example, start with the four Aces on top. Is there some combination of **in** and **out** shuffles that places the Aces to be every fifth card? Here is a dazzling discovery of Alex Elmsley, a British computer scientist. Consider the problem of bringing the original top card (at position 0) to position p by perfect shuffles. Elmsley observed that expressing p in binary, interpreting '0' as an **out** shuffle and '1' as an **in** shuffle does the job, irrespective of the deck size. For example, to bring 0 to 6, write 6 = 110 and perform **in**, **in**, **out**. If you try this with actual cards, remember that we start with 0, so that 6 is the position of the *seventh* card from the top. Since most of our readers are not accomplished card handlers, we later describe easy-to-do variants involving **inverse** shuffles so that the reader can follow along with cards in hand.

It is natural to try to solve the inverse problem: Is there a sequence of shuffles that brings a card at position p to the top? This turns out to be more difficult. Indeed, Elmsley, writing in the June 1957 issue of the British magic journal *Pentagram* writes: "I have so far been unable to discover a comparatively simple way of bringing a card to the top of a deck that is not a power of 2, e.g., 52. The only method I have found is much too complicated for practical use." Over the past 50 years, magicians and recreational mathematicians have studied 'Elmsley's Problem.' There have been special charts published giving the shortest sequence for various deck sizes. Recently, computer programs have been written and sold for doing the job.

In this article, we give a motivated development of our solution, and give a formula for the shortest sequence of **in** and **out** shuffles required to bring a card at position p to position q. We conclude this introduction with a brief description (and example) of our algorithm.

Algorithm to bring a card at position p to position 0. Working with a deck of $2n$ cards, define r by $2^{r-1} < 2n \le 2^r$ (so if $2n = 52$ then $r = 6$). For $0 < p < 2n - 1$, let $t = \lfloor (p+1)2^r / 2n \rfloor$ where $\lfloor x \rfloor$ denotes the largest integer less than or equal to x. For $p = 0$, set $t = 0$; for $p = 2n - 1$, set $t = 2r - 1$. Express t in binary as $t = t_{r-1}t_{r-2}\ldots t_1 t_0$ (with $t_i = 1$ or 0). Define "correction terms" $s = 2nt - 2^r p = s_{r-1}s_{r-2}\ldots s_1 s_0$ (with $s_i = 0$ or 1). The shuffling sequence is $t_{r-1} + s_{r-1}, t_{r-2} + s_{r-2}, \ldots, t_0 + s_0$, where each sum is in binary (without carries) with 1 as **in** and 0 as **out**. Any trailing 0s can be deleted.

Example. $2n = 52, p = 35$. Then r = 6,

$$t = \left\lfloor \frac{(p+1)2^r}{2n} \right\rfloor = \left\lfloor \frac{36 \cdot 64}{52} \right\rfloor = 44 = 101100$$

and $s = 2nt - 2^r p = 2288 - 2240 = 48 = 110000$. Now, the coordinatewise sum of 101100 and 110000 is 011100 which translates to **out**, **in**, **in**, **in** (the final two **out** shuffles do nothing to the top card). Further examples can be found throughout this article.

Note that when the deck size $2n$ equals 2^r, then the correction term $s = 0$, i.e., all the $s_i = 0$. In this case, *no* corrections are needed, which explains why decks of these sizes are especially nice.

Basic Properties of Perfect Shuffles

In this section we introduce basic properties of two permutations: the **in** and **out** shuffles. Throughout, we work with a deck of $2n$ cards, with positions labeled from 0 (top) to $2n - 1$ (bottom).

An **out** shuffle **O** is the permutation that sends the card in position i to position $2i - 1 \pmod{2n-1}$, for $0 \le i < 2n - 1$, and keeps the bottom card on the bottom. This can be represented as

$$\mathbf{O}(i) = \begin{cases} 2i \pmod{2n-1} & \text{if} \quad 0 \le i < 2n-1, \\ 2n-1 & \text{if} \quad i = 2n-1. \end{cases} \tag{1}$$

For example, after an **out** shuffle, 10 cards initially in positions 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 are now in positions 0, 2, 4, 6, 8, 1, 3, 5, 7, 9 (see Figure 1).

Similarly, for an **in** shuffle **I**,

$$I(i) = 2i + 1 \pmod{2n+1},\ 0 \le i \le 2n-1. \tag{2}$$

For example, after an **in** shuffle, 10 cards in positions 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 now are in positions 1, 3, 5, 7, 9, 0, 2, 4, 6, 8.

Figure 2. in shuffle of a ten card deck.

An **out** shuffle fixes the top and bottom cards and it is easy to see that the rest of the cards mix as an **in** shuffle on a deck of size two fewer. It follows that properties of repeated **in** shuffles can be understood via properties of repeated **out** shuffles. For example, consider the following problem: After how many **out** shuffles will a deck of size $2n$ recycle. If **O** stands for an **out** shuffle, let $\mathbf{O}^2$ stand for two **out** shuffles. From (1) above, there is a simple formula:

$$\mathbf{O}^2(i) = \begin{cases} 4i \pmod{2n-1} & \text{if} \quad 0 \le i < 2n-1, \\ 2n-1 & \text{if} \quad i = 2n-1. \end{cases} \tag{3}$$

Similarly, if $\mathbf{O}^k$ stands for k repeated **out** shuffles,

$$\mathbf{O}^k(i) = \begin{cases} 2^k i \pmod{2n-1} & \text{if} \quad 0 \le i < 2n-1, \\ 2n-1 & \text{if} \quad i = 2n-1. \end{cases} \tag{4}$$

The deck returns after k shuffles if and only if $\mathbf{O}^k(i) = i \pmod{2n-1}$, for $0 \le i < 2n-1$. From (4), this happens if and only if $2^k \equiv 1 \pmod{2n-1}$. The least such $k \ge 1$ is called the *order* of 2 (mod $2n-1$). For example, when $2n = 52$, $2n-1 = 51$, and the successive powers of 2 (mod 51) are 2, 4, 8, 16, 32, 13, 26, 1 that is, $2^8 = 256 \equiv 1 \pmod{51}$, or $ord_2(51) = 8$, so eight **out** shuffles recycle the deck. Likewise, eight **in** shuffles recycle a deck of size 50. The reader may show (by direct calculation or Fermat's little theorem) that $2^{52} \equiv 1 \pmod{53}$. Further, 2 is a *primitive root* modulo 53 (so 52 is the least such power). This says that a deck of size 54 recycles after 52 outshuffles, or equivalently, when ignoring the top and bottom of these 54 cards, that a deck of size of 52 recycles after 52 **in** shuffles. We are embarrassed to report that we first learned this by actually shuffling the cards.

Is there a pattern for the number of shuffles needed to recycle? Here are the numbers for decks of sizes 2 to 54.

deck size $2n$	2	4	6	8	10	12	14	16	18	20	22	24	26	
$ord_2(2n-1)$	1	2	4	3	6	10	12	4	8	18	6	11	20	
deck size $2n$	28	30	32	34	36	38	40	42	44	46	48	50	52	54
$ord_2(2n-1)$	18	28	5	10	12	36	12	20	14	12	23	21	8	52

Table 1. Values of $ord_2(2n-1)$ for deck sizes $2n$. These values give the number of **out** shuffles needed to recycle a deck of size $2n$ and the number of **in** shuffles needed to recycle a deck of size $2n - 2$.

Is there a pattern in these numbers? The first surprise is that the numbers are not increasing. Larger decks can recycle after fewer shuffles. Thus, a 16 card deck recycles after 4 **out** shuffles while a 14 card deck requires 12 **out** shuffles. Decks of size 2^k recycle after k **out** shuffles, since $2^k \equiv 1 \pmod{2^k - 1}$. Aside from these, very little is known; the order of 2 is one of the mysteries of mathematics. Consider the question of whether there are arbitrarily large integers $2n$ such that 2 is a primitive root modulo $2n - 1$. This is a famous conjecture of E. Artin. A well-studied, and completely intractable, problem! It is known to be true if the Generalized Riemann Hypothesis is true. We find it tantalizing that simple questions about shuffling cards can lead to questions well beyond modern mathematics.

Returning to Elmsley's problem, we initially thought it too would be intractable, roughly equivalent to the problem of "finding logarithms in finite fields." This last problem lies at the root of several widely used cryptographic protocols. After all, if there were a simple rule, someone should have found it in over 50 years of fooling around.

In *The Mathematics of Perfect Shuffles*, the authors, working with Bill Kantor, determined the structure of the group $\langle \mathbf{I},\mathbf{O} \rangle$ generated by arbitrary **in** and **out** shuffles. From this work it follows (and, as shown below, is easy to see directly), that for any p and q, there is a sequence of **in** and **out** shuffles sending the card in position p to position q. These proofs give no insight into the *shortest* way. In fact, our arguments below will settle this problem by finding **all** perfect shuffle sequences that move card p to position q.

Before proceeding, it will be useful to introduce the two inverse shuffles $\mathbf{O}^{-1}$ and $\mathbf{I}^{-1}$. These "undo" or "unshuffle" the results of **O** and **I**. They can easily be carried out with a deck of cards in hand. Hold the cards face down as if dealing in a card game. Deal the cards face up, alternately, into two piles, dealing left, right, left, right, ..., until all the cards have been dealt.

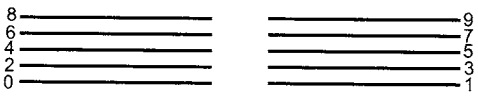

Figure 3. Ten cards dealt alternately into two face-up piles.

To complete $\mathbf{O}^{-1}$, place the right pile on the left pile (so that the original top and bottom cards remain on the bottom and top, respectively). To complete $\mathbf{I}^{-1}$, pick up the piles the other way (so that the original top and bottom cards wind up in the middle). In both cases, finish by turning the deck face down. We can write a formula for $\mathbf{O}^{-1}$ and $\mathbf{I}^{-1}$ as permutations of the set $\{0, 1, 2, \ldots, n-1\}$ by:

$$\mathbf{O}^{-1}(i) = \begin{cases} \left\lfloor \frac{i}{2} \right\rfloor & \text{if } i \text{ is even,} \\ \left\lfloor \frac{i}{2} \right\rfloor + n & \text{if } i \text{ is odd,} \end{cases} \tag{5}$$

$$\mathbf{I}^{-1}(i) = \begin{cases} \left\lfloor \frac{i}{2} \right\rfloor + n & \text{if } i \text{ is even,} \\ \left\lfloor \frac{i}{2} \right\rfloor & \text{if } i \text{ is odd.} \end{cases} \tag{6}$$

As mentioned earlier, $\lfloor x \rfloor$ denotes the largest integer less than or equal to x.

If **out** shuffles recycle after k and **in** shuffles recycle after j, then $\mathbf{O}^{-1} = \mathbf{O}^{k-1}$ and $\mathbf{I}^{-1} = \mathbf{I}^{j-1}$, so any arrangements achievable by **outs** and **ins** are achievable by $\mathbf{O}^{-1}$ and $\mathbf{I}^{-1}$, and vice versa.

Let us argue that there is some sequence of **O**s and **I**s that brings p to q. First, bring p to the top by a sequence of $\mathbf{O}^{-1}$s and $\mathbf{I}^{-1}$s as follows: Deal into two piles, then placing the pile *not* containing the card labeled p onto the other pile. Thus, each time with the cards turned back face down, the card labeled p is in the top half. If this is repeated, then after at most r times, the card labeled p comes to the top (here, $2^{r-1} < 2n \le 2^r$). From here, the current top card (labeled p) can be brought to position q using Elmsley's binary procedure explained in the introduction. This may give a long sequence of shuffles, but at least it shows it can be done. Next, we show how to bring any card to the top efficiently.

Bringing any card to the top

In this section we solve Elmsley's problem by finding a succinct way of determining a sequence of **in** and **out** shuffles that brings a card at position p to position 0. Since the equations for the two shuffles involved different moduli (Eqs. (1), (2)), this seems like a messy problem. The key is to work with inverse shuffles. These involve dividing by 2, taking the floor, and perhaps adding n. The decision to add n or not depends on the parity of i. We first disregard this issue of parity and put it back at the end, as a correction term. Throughout, we make constant use of the identity $\lfloor \lfloor x \rfloor /2 \rfloor = \lfloor x/2 \rfloor$.

Step One: Building a Tree. Form a labeled binary tree $T(2n)$ with $r+1$ levels, where $2^{r-1} < 2n \le 2^r$, as follows: The root v_0 is at level 0 and labeled with $\lambda(v_0) = 0$. In general, if v is a vertex of $T(2n)$ at level i which has been labeled $\lambda(v) = m$, the two 'children' of v (on level $i+1$) will have labels $\lfloor m/2 \rfloor$ and $\lfloor m/2 \rfloor + n$. Write this as $\lfloor m/2 + t_i n \rfloor$, where $t_i = 0$ or 1, for $0 \le i \le r$. We illustrate this in Figure 4.

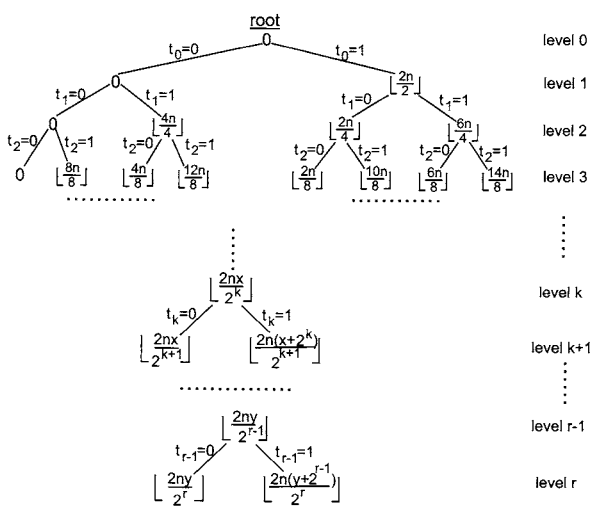

Figure 4. The tree $T(2n)$.

If $t = \sum_{i=0}^{r-1} t_i 2^i$, the value of the leaf v_t corresponding to the choices $(t_0, t_1, \ldots, t_{r-1})$ going down from the root is $\lambda(v_t) = \lfloor 2nt/2^r \rfloor$. More generally, it can be seen by induction that the value assigned to vertex v at level k corresponding to the choice $(t_0, t_1, \ldots, t_{k-1})$ is $\lambda(v) = \lfloor 2nt(k)/2^k \rfloor$ where

$$t(k) = \sum_{i=0}^{k-1} t_i 2^i.$$

Step Two: Relating the tree to shuffling. For $t = \sum_{i=0}^{r-1} t_i 2^i$, if we write $2nt = \sum_{i\geq 0} s_i 2^i$ in binary, then on one hand,

$$\left\lfloor \frac{2nt}{2^k} \right\rfloor = \left\lfloor \sum_{i\geq k} s_i 2^{i-k} \right\rfloor = \ldots s_{k+2} s_{k+1} s_k$$

in binary. On the other hand,

$$\begin{aligned} \left\lfloor \frac{2nt}{2^k} \right\rfloor &= \left\lfloor \frac{2n}{2^k} \left(\sum_{i=0}^{k-1} t_i 2^i + \sum_{i\geq k} t_i 2^i \right) \right\rfloor \\ &= \left\lfloor \frac{2n}{2^k} (t(k) + 2^k X) \right\rfloor \\ &= 2nX + \left\lfloor \frac{2nt(k)}{2^k} \right\rfloor \end{aligned}$$

for some integer $X \geq 0$. Hence, the *parity* of $\lfloor 2nt(k)/2^k \rfloor$ is just s_k. This can be used to determine if the choice taken at any vertex is $\mathbf{I}^{-1}$ or $\mathbf{O}^{-1}$.

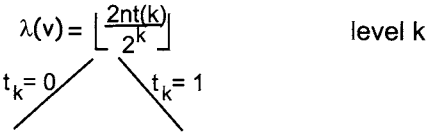

Figure 5. A general branch.

Specifically, if the value $\lfloor 2nt(k)/2^k \rfloor$ is *even* then $t_k = 0$ corresponds to $\mathbf{O}^{-1}$ and $t_k = 1$ corresponds to $\mathbf{I}^{-1}$. On the other hand, if $\lfloor 2nt(k)/2^k \rfloor$ is *odd*, then $t_k = 0$ corresponds to $\mathbf{I}^{-1}$ and $t_k = 1$ corresponds to $\mathbf{O}^{-1}$ (see Eqs. (5), (6) and Figure 5).

The shuffle at a vertex is determined by $u_k = s_k + t_k \pmod 2$ where $u_k = 0 \leftrightarrow \mathbf{O}^{-1}$ and $u_k = 1 \leftrightarrow \mathbf{I}^{-1}$.

Step Three: Putting the pieces together. We want to find a shuffle sequence of $\mathbf{O}^{-1}$s and $\mathbf{I}^{-1}$s that will bring the top card to position p. Thus, set

$$\left\lfloor \frac{2nt}{2^r} \right\rfloor = p.$$

This implies

$$\frac{2^r p}{2n} \leq t < \frac{2^r (p+1)}{2n}. \qquad (7)$$

Since $2^{r-1} < 2n \leq 2^r$, then for any p, there is always at least *one* and at most *two* integers t satisfying (7). In particular, if we expand $(p+1)/2n$ (base 2) as

$$\frac{p+1}{2n} = .\alpha_1 \alpha_2 \alpha_3 \ldots$$

then we can choose $t = \sum_{i=1}^{r} \alpha_i 2^{r-i} = \alpha_1 \alpha_2 \ldots \alpha_r$ (base2). Equivalently, we can choose $t = \lfloor (p+1)2^r / 2n \rfloor$ for $0 < p < 2n-1$.

For $p = 0$, take $t = 0$, and for $p = 2n - 1$, take $t = 2^r - 1$. For convenience, we let

$$s = 2nt - 2^r p = s_{r-1} s_{r-2} \ldots s_1 s_0 \quad \text{(base 2)}.$$

Now, with $2nt = \sum_{i\geq 0} s_i 2^i$, the s_i, $0 \leq i \leq r-1$ provide corrections to the t_i to determine which of $\mathbf{O}^{-1}$ or $\mathbf{I}^{-1}$ is carried out at each stage. The final result is summarized in the algorithm given in the Introduction.

Remarks

(i) The description above determines inverse shuffles to bring 0 to p. It must be read 'left to right' to determine the sequence of **in** and **out** shuffles to bring the card at position p to the top.

(ii) Example. If $2n = 52, p = 36$, then

$$r = 6,\ t = \left\lfloor \tfrac{37 \cdot 64}{52} \right\rfloor = 45 = 101101 \text{ (base 2)}$$
$$s = 2340 - 2304 = 36 = 100100 \text{ (base 2)}.$$

Thus, u = 001001 and **OOIOOI** (read left to right) is the desired sequence.

(iii) As we remarked earlier, there may be *two* values of t satisfying (7). In this case one shuffle sequence will have length r and one will have length less than r (and will be the shortest shuffle sequence bringing p to the top). For example, take $2n = 52, p = 30$. Then, $r = 6$, and since $64 \cdot 30 / 52 \leq 37, 38 < 64 \cdot (30 + 1)/52$, we may use $t = 37$ or $t = 38$. For $t = 37 =$ 100101, $s = 1924 - 1920 = 4 = 000100$, $u = 100001$, which results in the shuffle sequence **IOOOOI**. For $t = 38 = 100110$, $s = 1976 - 1920 = 56 = 111000$, $u = 011110$, which results in the shuffle sequence **OIIIIO**, which can be truncated to **OIIII**. Thus, in fact, only 5 shuffles are needed to bring the card at position 30 to the top, and this is the minimum possible.

More generally, the algorithm shows that *any* card can be effectively brought to the top in *at most* r shuffles, with $2^{r-1} < 2n \leq 2^r$. As we have seen, there can be two different shuffle sequences that accomplish this, although one of them will require r shuffles and one will need fewer than r shuffles. For example, this is always the case when $2n$ exactly divides $2^r p$ (provided $2n < 2^r$), since in this case we can choose t to be either $2^r p/2n$ or $(2^r p/2n) + 1$.

(iv) From the analysis above, we see that the 2^{r-1} possible perfect shuffle sequences of length $r - 1$ leave *different* cards on top. This fails for shuffle sequences of length r unless $2n = 2^r$. Alex Elmsley has exploited this in a special case. Working with a packet of 8 cards (arranged in a known order), he has the spectator deal into two piles (say, face up), and choose to remember either top card, and drop the other packet on the noted card. The cards are turned down and this is repeated with two other spectators. Because of the uniqueness, the current

top card determines all three selections. Marvelous presentations and variations are explained on pages 80-88 of Vol. II of Elmsley's *Collected Works*.

Moving card *p* to position *q*

The reasoning behind basic properties of perfect shuffles allows explicit determination of the shortest sequence of shuffles to bring the card initially at position p to position q. Form a tree $T_q(2n)$ with labeled vertices, similar to the tree $T(2n)$ in Figure 4, but with the root labeled by q instead of 0 (see Figure 6).

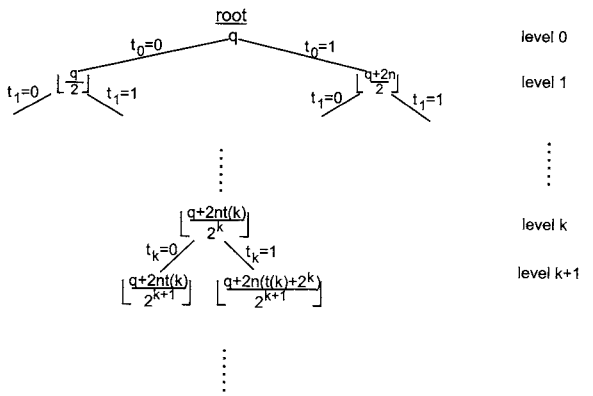

Figure 6. The tree $T_q(2n)$.

Now, we need to find $t = t_{r-1} \ldots t_1 t_0$ (with $t_i = 0$ or 1), to satisfy

$$\left\lfloor \frac{q+2nt}{2^r} \right\rfloor = p.$$

Thus,

$$\frac{2^r p - q}{2n} \le t < \frac{2^r (p+1) - q}{2n}.$$

As before, such a t always exists and there are at most two such t's of length at most r. A correction term is computed as:

$$s = 2nt + q - 2^r p = s_{r-1} s_{r-2} \ldots s_1 s_0$$

with $s_i = 0$ or 1. Taking the mod 2 coordinatewise sums,

$$t_{r-1} + s_{r-1}, t_{r-2} + s_{r-2}, \ldots, t_1 + s_1, t_0 + s_0$$

and translating these values to **O**s and **I**s gives the desired sequence.

Example. Suppose $n = 52$, $p = 51$, $q = 1$. Thus, the bottom card is to be moved to second from the top. Here, $r = 6$ and $(64 \cdot 51 - 1)/52 \le t < (64 \cdot 52 - 1)/52$. Thus, $t = 63 = 111111$. Next, $s = 3277 - 3264 = 13 = 001101$. Taking the mod 2 sum of t and s gives the final shuffle sequence **IIOOIO**. Now, of course, we are not at liberty to delete the trailing **O** since an **out** shuffle does *not* preserve the position of card q, unless q is 0 or $2n-1$.

The analysis above may be used to determine some of the "geometry" of the shuffle group $\langle \mathbf{I}, \mathbf{O} \rangle$ using **I** and **O** as generators. For example, suppose we want to determine all possible cycles of length m. For this, set $p = q$ in the preceding analysis. Thus,

$$\left\lfloor \frac{2nt+q}{2^m} \right\rfloor = q, \quad \left\lfloor \frac{2nt-(2^m-1)q}{2^m} \right\rfloor = 0, \quad \frac{2nt}{2^m-1} - 1 \le q \le \frac{2nt}{2^m-1},$$

where $0 \le t \le 2m-1, 0 \le q \le 2n-1$. It is exactly these values of q for which m-cycles exist using the card in position q.

Example. Take $2n = 52$, $m = 5$. The 32 possible values of q are:

$$0,1,3,5,6,8,10,11,13,15,16,18,20,21,23,25,26,$$
$$28,30,31,33,35,36,38,40,41,43,45,46,48,50,51.$$

Thus, there are two trivial 1-cycles (**O** preserves the top and bottom cards). The remaining 30 values break up into six genuine 5-cycles. For instance, for $q = 13$, $t = 8 = 01000$ and $s = 52 \cdot 8 - 31 \cdot 13 = 13 = 01101$. Thus, adding t and s coordinatewise mod 2, we get **OOIOI**, which gives

$$13 \xrightarrow{O} 26 \xrightarrow{O} 1 \xrightarrow{I} 3 \xrightarrow{O} 6 \xrightarrow{I} 13.$$

Research projects and final remarks

Much of the analysis above can be carried over to more complex mixing schemes involving dealing more piles. Consider dealing a deck of $3n$ cards face up, one at a time, into three piles. Now the piles can be picked up left to right or right to left. Determining just what can be done and how to do it is a research project.

It may be possible to determine the diameter of the shuffle group using the considerations above. We believe the group is doubly transitive on the pairs $\{0, 2n-1\}, \{1, 2n-2\}, \ldots, \{n-1, n\}$. Determining the shortest shuffling sequence bringing two cards to two given positions may determine the shortest shuffling sequence to bring any possible arrangement to any other.

A related topic is to use the results on the basic properties of perfect shuffles with $k > r$ to study the case when there are many ways to bring the card at position p to the top. What is the structure of these? What can be done? ■

Further Reading

A history and extensive development is given in "The Mathematics of Perfect Shuffles," *Adv. in Appl. Math* (1983), 175–196, by P. Diaconis, R. L. Graham, and W. Kantor. For a very nice, closely related paper see "Moving card i to position j with perfect shuffles" by Sarnath Ramnath and Daniel Scully,

Mathematics Magazine 69 (1996), 362-365. See also "Groups of Perfect Shuffles," *Mathematics Magazine*, (1987), 3–14, by S. Medvedoff and K. Morrison, and "Unshuffling for the Imperfect Magician," *Math Horizons* (2004), 13–16, by D. Ensley. Practical instructions for card tricks, how to perform an actual triple shuffle, and applications to computer science are given in *Magic Tricks, Card Shuffling, and Dynamic Computer Memories* by S. Brent Morris, published by MAA.

Acknowledgment

As this paper was being completed, we learned of the death of Alex Elmsley. We dedicate it to his memory. This research was supported in part by NSF Grants DMS 0505673 and CCR-0310991.