**ArsDigita University**
**Month 2 – Discrete Mathematics and Probability**

**Professor Shai Simonson**

# The Combinatorial Card Trick – Challenge Problem

We showed a method in class based on combinatorics, which allows a magician and an accomplice to perform the following trick. The magician asks someone to choose five cards randomly from a standard deck. The accomplice then looks at these cards, and shows four of them in a particular order to the magician, who then immediately identifies the last card.

At first thought the trick seems impossible, because there are only 4! ways to order four cards, and 24 pieces of information is not enough to identify a unique value among the possible 48 cards remaining. However, a more careful analysis reveals that the accomplice has more choices than 24 up his sleeve. The accomplice may choose which four cards to show, as well as their order. There are C(5,4) ways of choosing four cards from five, hence the accomplice has 120 different pieces of information he can send. The hard part is that the magician cannot easily decode which of these 120 was sent. She can easily decode 24 pieces of information, but not so easily the extra factor of five.

One method that allows the magician to decode more than the obvious 24 is to notice that among the five cards chose, there must be two of the same suit (due to the pigeonhole principle). The first card shown by the accomplice is one of these two cards, and the second is never shown. If we number the cards in a suit from 1 to 13, and do arithmetic modulo 13, then given any two cards, there is always one which is six or less below the other (again the pigeonhole principle – if they were both 7 or more less than the other, then there needs to be 14 cards). The accomplice chooses the card that is 6 or less below the other (modulo 13). For example, given the 3 and the Jack, we choose the Jack.

The accomplice then chooses the order of the next three cards to encode a number from 1 to 6. The magician decodes this number the last three cards, (value 1-6) and adds it to the first card, in order to recover the identity of the missing card.

There are many ways to identify the 3! permutations of three cards with the numbers 1 through 6. We do it in a way that allows fast decoding. The position of the lowest card tells us 1, 2 or 3. The order of the higher two cards, tells us whether or not to add 3 to this number: if they are in order, do not add 3 otherwise add 3. For example, the sequence 3D AC 2H decodes to the number 5. This is because the smallest value is the A, which is in position two, and the remaining two cards 3D 2H are not in order, so we add 3 giving 5. For the purposes of ordering we assume that the cards from lowest to highest are A through K, and if there is a tie then we break it by the order of the suits which is Clubs, Diamonds, Hearts, and Spades.

For example, if the four cards seen by the magician are in order: 4D, 9S, JD, 9H then the hidden card is the 4D + 3 = 7D. We add three because the smallest card of the last three cards is the 9H, and the remaining two cards 9S and JD are in order (giving 3 + 0).

## The Challenge

The question you must explore is whether we can do this trick with a larger deck of cards.

It seems that the ideal case would send all 120 pieces of information from the accomplice to the magician. This would allow the trick to be done with 124 cards! It would be impossible to do the trick with 125 cards or more, because not enough unique messages are available to the accomplice.

Another way to look at this, is that the accomplice has $C(52, 4) \times 4!$ ways to send his message, and the number of possible sets of cards chosen is $C(52, 5)$. The first number is about 2.5 times larger than the second number, hence there is some chance of success. Note that when the deck has more than 124 cards, say 125, the first number is $C(125, 4) \times 4!$, which is strictly smaller than the second, $C(125, 5)$. When this happens, no strategy is possible. If one were, then by the pigeonhole principle, there would exist two sets of five cards, for which the accomplice is sending the same ordered subset of four cards. But if that is the case, then the magician cannot possibly determine which card is hidden.

Our strategy seems to have no slack at all. We divide the deck into exactly four groups to guarantee the duplicate suit, and choose the first card and hidden card to get the number of possible hidden cards down to six. Then we have six pieces of information left (three ordered cards) which we can use to identify the hidden card – just enough. It seems unlikely that we could extend our strategy to work for 124 cards.

Maybe there is a better strategy (one that perhaps does not have an easily decodable scheme) – that might work for a larger deck?

In order to explore this question, let's think like engineers, and look at our boundary conditions, consider special cases and the extreme cases.

Lower Bounds on the Number of Cards Possible – Based on Our Working Strategy

1. If we do the trick by letting the person choose 4 cards and the accomplice shows an ordered subset of three of them, then what is the maximum number of cards for which our strategy works?
2. Same question for 3 cards? 2 cards?
3. Let's go in the other direction. If we let the person choose 6 cards and the accomplice shows an ordered subset of 5, then how large a deck can our strategy handle?

4. Generalize your results above for the case where $n$ is the number of cards chosen, and the accomplice chooses an ordered subset of $n-1$.

Upper Bounds on the Number of Cards – Based on a Combinatorial Idea

5. The upper bound for $n=5$ is 124 cards. There are two arguments above explaining the reason for this. What are the upper bounds for $n = 2, 3, 4$?
6. Write a formula for the upper bound in terms of $n$.

Now let's analyze the possibility of other strategies. In practice, a strategy needs to be easily decodable, but theoretically a strategy is no more or less than list of what the accomplice should do in every possible situation. As long as this function is one-one and onto, then the magician can reverse it, and look up the appropriate hidden card, with a computer program if necessary.

7. Let $n$ be the number of cards chosen, where the accomplice chooses an ordered subset of $n-1$. Calculate formulas in terms of $n$, for
    a. The number of ways the contestant can choose his cards.
    b. The number of different permutations available to the accomplice for each of the contestant's choices. (For $n=5$, there are 120 possibilities).
    c. The number of different *strategies* available to the accomplice and magician. (A *strategy* can be defined as a choice of permutation for each of the possible set of cards in the contestant's hand.)

Given a strategy, and a set of $n$ cards, we can calculate exactly which ordered set of $n-1$ cards are shown. A collection of these ordered sets can be enumerated one by one for each of the possible choices of $n$ cards. A strategy is *successful* whenever this set contains no duplicates. If there was a duplicate then the magician would have no way to know which of the two different hidden cards to decode. That is, the encoding function used by the accomplice, must be one-one and onto.

8. Let $n=3$, and the strategy be our original strategy.
    a. Using the maximum size deck you calculated in problem 2, enumerate the actual ordered sets for each of the possible selections of 3 cards. Verify that the strategy is successful.
    b. Add one card to the size of the deck in part a., and again enumerate the actual ordered sets for each of the possible selections of 3 cards. Verify that this time the strategy is unsuccessful.
9. Let $n=3$. By hand, explicitly construct a strategy that is successful for a deck whose size is one greater than the maximum.

This proves that it is possible to improve on our strategy. The question is: how much larger of a deck can we handle? Your challenge is to determine how large a deck can be managed for a given $n$. This involves a great deal of computation, and you will almost surely need to resort to the aid of a computer.

A brute force approach is to fix *n*, and initialize the size of the deck to one plus the maximum size that works for our initial strategy. Enumerate all possible strategies, and stop when one is successful. If no strategies are successful, then stop and print out the size of the previous deck. If your program just keeps running because of a combinatorial explosion, then you can use the last printed deck size as your best known bound.

10. The challenge is to fill in the table below, which relates the number of cards chosen from the deck, to the size of the deck that can be handled by our original strategy (lower bound), the theoretical maximum size of the deck for any strategy (upper bound), and the largest size deck that can be handled with some strategy.

| | Size of the Deck for which the Trick Can be Done | | |
|---|---|---|---|
| Number of Cards Chosen | Original Strategy | Best Known | Upper Bound |
| 2 | | | |
| 3 | | | |
| 4 | | | |
| 5 | | | |
| … | | | |
| *n* | | | |

Does the best strategy you can find ever reach the upper bound? I will give $25 for a complete formula relating the number of cards to the correct least upper bound.