Let's Play Slap!

This week's Fiddler on the Proof (13 October 2023) asks:

Every so often, I play the card game Slap with someone who really doesn't like to lose. (You know who you are.) But before we get into all that, here are the rules of the game:

- Two players are both dealt 26 cards from a standard deck.
- From there, they take turns putting down one card at a time. Let's interchangeably designate the players as "A" and "B."
- If A puts down a jack, then B puts down *one* card. If it's a face card, the game continues with A. Otherwise, A wins the round and adds those cards to the bottom of their pile so that the last card played is at the very bottom of the pile.
- If A puts down a queen, B has *two* chances to put down a face card, at which point the game would continue with A. Otherwise, A wins the round and collects the cards.
- If A puts down a king, B has *three* chances to put down a face card, at which point the game would continue with A. Otherwise, A wins the round and collects the cards.
- If A puts down an ace, B has *four* chances to put down a face card, at which point the game would continue with A. Otherwise, A wins the round and collects the cards.
- If, at any point, two cards have been played in succession (by different players or the same player) that have the same number or face, then both players can slap the pile of played cards to win the round and collect the cards.
- Whoever won the previous round plays the first card in the next round.

Now look. I really like this opponent and I don't want to hurt their feelings. And so, while shuffling, I rig the deck so that they get a majority of the jacks, a majority of the queens, a majority of the kings, and a majority of the aces.

Nobody likes a cheater, and so my opponent rightfully accuses me of rigging the game. In my defense, I say that the probability of one player randomly receiving the majority of each kind of face card is reasonable. What is this probability?

52 Slots

What are the chances that 52 randomly shuffled cards will put at least 3 J's, 3 Q's, 3 K's, and 3 A's in one predetermined half of the deck?

The way I tackled this was to imagine 52 empty slots for the cards. Half of the slots are labeled "A" and the other half of the slots are labeled "B". Then I imagined randomly placing the face cards one at a time into the open slots.

Let's start with the J's. The first J has a 26/52 chance of being placed in an A slot and a 26/52 chance of being placed in a B slot.

The second J has either 25/51 chance of being placed in an A slot or a 26/51 chance of being placed in an A slot depending on what happened with the first J.

As we continue placing J's, and then the other face cards, the situation gets more involved as the probability for each new card depends on where the previous cards landed.

The best way I could think of to keep track of all the probabilities and combinations was with a recursive function to calculate the probability that a quadruple majority of face cards ended up in the A slots.

The following function calculates this probability. Here is the meaning of the parameters and variables it uses:

- aSlots is the number of A slots that are still open.
- bSlots is the number of B slots that are still open.
- setsToGo is the number sets of different face cards yet to be slotted. (Initially 4, for the jacks, queens, kings, and aces.)
- need is the number of the current face card yet to be slotted that need to be slotted in A slots to achieve a majority for that face card
- outOf is the total number of the current face card yet to be slotted
- probOfA is the probability that the next card will land in an A slot
- probAfterA is the probability of slotting all remaining cards to achieve the required majority after a card lands in an A slot
- probOfB is the probability that the next card will land in a B slot
- probAfterB is the probability of slotting all remaining cards after a card lands in a B slot

```
func prob( aSlots:Int, bSlots:Int, setsToGo:Int, need:Int, outOf:Int ) -> Double
    {
    if need == 0
        {
        if setsToGo == 0 { return 1.0 }
        return prob( aSlots:aSlots, bSlots:bSlots,
                     setsToGo:setsToGo-1, need:3, outOf:4 )
        }
    if need > outOf { return 0.0 }

    let allSlots   = aSlots + bSlots
    let probOfA    = aSlots.double / allSlots
    let probAfterA = prob( aSlots:aSlots-1, bSlots:bSlots,
                           setsToGo:setsToGo, need:need-1, outOf:outOf-1 )
    let probOfB    = bSlots.double / allSlots
    let probAfterB = prob( aSlots:aSlots, bSlots:bSlots-1,
                           setsToGo:setsToGo, need:need, outOf:outOf-1 )
    return probOfA * probAfterA + probOfB * probAfterB
    }
```

Invoking this function with the following parameters calculates the chance that at least 3 of each of the face cards will be placed in the A slots.

```
prob( aSlots:26, bSlots:26, setsToGo:4, need:0, outOf:0 )

0.0034147591571642524
```

That means the probability that one (prespecified) player gets a majority of each set of face cards is about a **third of a percent**.

Extra Credit

> It turns out that rigging the deck while shuffling is very hard to do. Instead, I decide to cheat in my opponent's favor by ensuring my reaction time is slower than theirs at every single slapping opportunity. In other words, I lose every slap.
>
> Assuming I shuffle and distribute the cards randomly, and that my opponent plays the first card in the game, what is my probability of winning?

Finding an exact answer to this question seems quite difficult. There are so many states in the game that keeping track of them all is unmanagable. Instead, I used computer simulation to get an approximate answer.

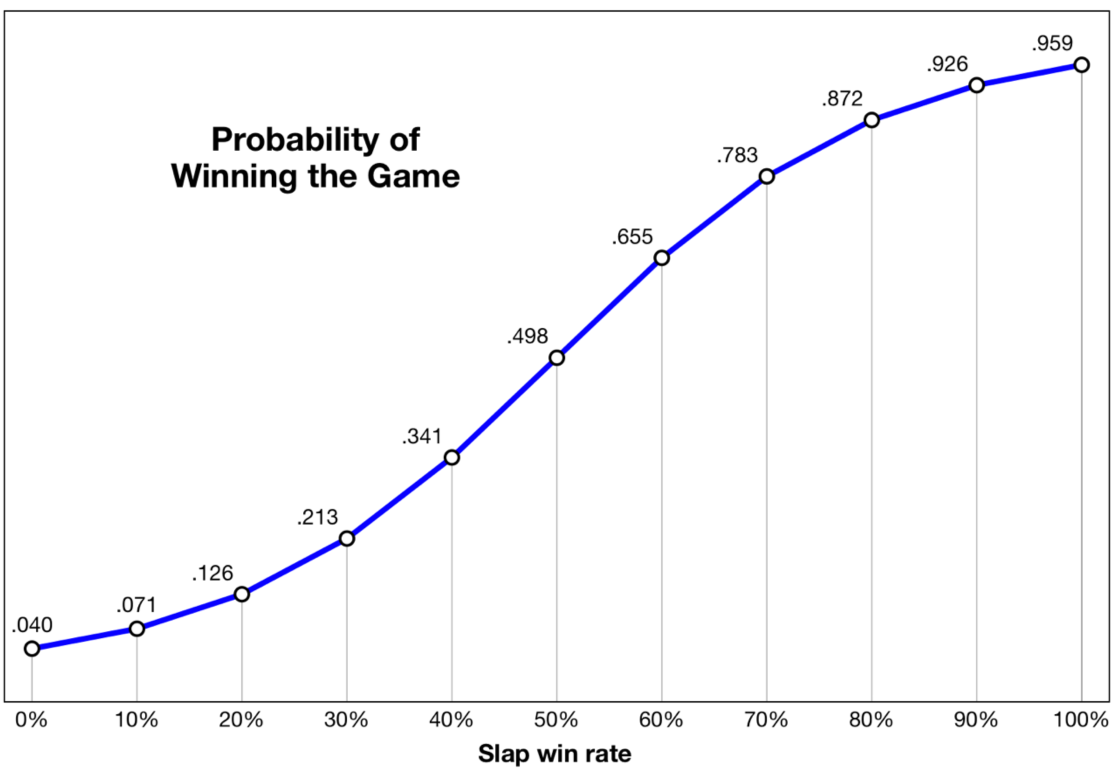
I coded up the game, making sure that player A went first each time and that A also won all the "slapping" contests. It turned out that player A won 958,743,054 of out of 1,000,000,000 games I ran, or about 95.8743% of the time.

"Your" (player B's) probability of winning is therefore about **4.1257%**. (Which is much less than the probability I made a mistake coding up the simulation!)

Converting Slap Rate to Win Rate

Suppose player A's probablity of winning each slapping contest is *p*. What are A's chances of winning the game?

I ran my simulation using different values of *p* and the results are summarized in the following chart.



Notice the values in this chart are not quite symmetric. This reflects that going first is a slight disadvantage.