Incredible Comebacks

This week's Fiddler on the Proof (7 June 2024) asks:

> Suppose you're playing a game in which there are five "possessions." For each possession, there's a 50 percent chance that *your* team scores one point. If you don't score, then your opponent instead scores one point.
>
> After the game, ESPN reports that your opponent's chances of winning were "75 percent chance or higher" *at some point* during the game (i.e., before the final possession is complete).
>
> Given this information, what was the probability that your team actually won the game?

Probability According to Score

The following chart shows our probability of winning the game based on the number of points we have versus our opponent.

Opponent's Points \ Our Points	0	1	2	3	4	5
0	0.5000	0.6875	0.8750	1.0000	1.0000	1.0000
1	0.3125	0.5000	0.7500	1.0000	1.0000	
2	0.1250	0.2500	0.5000	1.0000		
3	0.0000	0.0000	0.0000			
4	0.0000	0.0000				
5	0.0000					

The entries in red indicate that the game is over. The entries in yellow are those scores where our opponent has at least a 75% chance of winning the game.

Sequences

There are 32 possible sequences the game might take. Let's see which of them give our opponent at least 75% winning chances at some point. Also, which sequences end up winning the game for us.

Sequence	Opp reaches ≥ 75%	We win the game
LLLLL	✅	
LLLLW	✅	
LLLWL	✅	
LLLWW	✅	
LLWLL	✅	
LLWLW	✅	
LLWWL	✅	
LLWWW	✅	✅
LWLLL	✅	
LWLLW	✅	
LWLWL	✅	
LWLWW	✅	✅
LWWLL		
LWWLW		✅
LWWWL		✅
LWWWW		✅

Sequence	Opp reaches ≥ 75%	We win the game
WLLLL	✅	
WLLLW	✅	
WLLWL	✅	
WLLWW	✅	✅
WLWLL		
WLWLW		✅
WLWWL		✅
WLWWW		✅
WWLLL		
WWLLW		✅
WWLWL		✅
WWLWW		✅
WWWLL		✅
WWWLW		✅
WWWWL		✅
WWWWW		✅

A total of 16 sequences result in the opposition achieving 75% winning chances at some point during the game. Of those 16 sequences, only 3 result in our eventually winning the game.

$$\frac{3}{16} = 0.1875$$

If you ask the question the other way around, you get the same answer: "Knowing that we won the game, what are the chances that our opponent was ≥ 75% at some point?"

Extra Credit

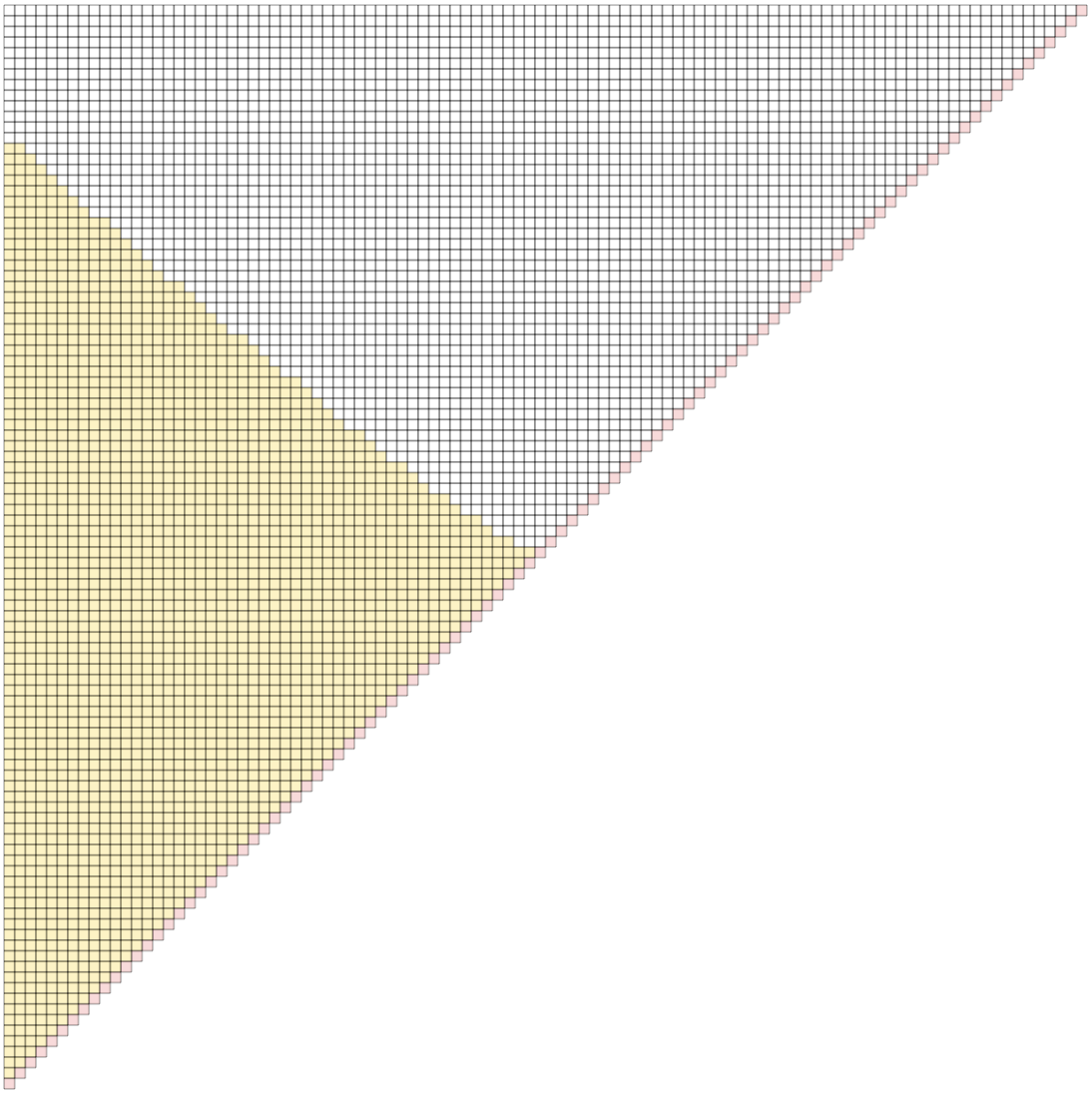
Instead of five possessions, now suppose there are 101. Again, with each possession, there's a 50 percent chance that your team scores one point. If you don't score, then your opponent instead scores one point.

After the game, ESPN reports that your opponent's chances of winning were "90 percent chance or higher" *at some point* during the game (i.e., before the final possession is complete).

Given this information, what was the probability that your team actually won the game?

Same idea, just bigger. And the threshold for coming back from is now 90 percent.

Here are the scores that give the opposition 90% chance or greater of winning the game:



I wrote a program to go through all the games that include a time when the opposing team was 90% or better to win. Of those games, I checked how many ended up being a win for us.

$$\frac{\text{comeback wins}}{\text{comeback attempts}} = \frac{1009509460763345607322194896 24}{1303940929380534277290406564216} \cong 0.07741987677639164$$

This time I got a different answer when looking at the question the other way round:
"Given that we won the game, what is the chance that we came back from $\leq 10\%$ chance of winning?"

$$\frac{\text{comebacks wins}}{\text{all wins}} = \frac{1009509460763345607322194896 24}{1267650600228229401496703205376} \cong 0.07963625470469483$$

```
let N = 101

// Matrices to hold already-computed values
var gameCountMatrix = Matrix<BigInt?>( size:(N+1,N+1), zero:nil )
var winCountMatrix = Matrix<BigInt?>( size:(N+1,N+1), zero:nil )
var attemptsBeforeWorthyMatrix = Matrix<BigInt?>( size:(N+1,N+1), zero:nil )
var comebacksBeforeWorthyMatrix = Matrix<BigInt?>( size:(N+1,N+1), zero:nil )

struct GameScore
    {
    var us, them : Int      // Current score of game

    var isWorthy : Bool     // True if this score is now worthy of a comeback
        {
        assert( self.isGameOver.not )
        return self.winCount() * BigInt(10) <= self.gameCount()
        }
    var isGameOver : Bool { return self.us + self.them >= N }
    var didWin : Bool { return self.us > self.them }
    var afterWin  : GameScore { var score = self ; score.us   += 1 ; return score }
    var afterLoss : GameScore { var score = self ; score.them += 1 ; return score }

    // Number of games possible from this score
    func gameCount() -> BigInt
        {
        if let answer = gameCountMatrix[us,them] { return answer }
        if self.isGameOver { return 1 }
        let answer = self.afterWin.gameCount() + self.afterLoss.gameCount()
        gameCountMatrix[us,them] = answer
        return answer
        }

    // Number of wins possible from this score
    func winCount() -> BigInt
        {
        if let answer = winCountMatrix[us,them] { return answer }
        if self.isGameOver { return self.didWin ? 1 : 0 }
        let answer = self.afterWin.winCount() + self.afterLoss.winCount()
        winCountMatrix[us,them] = answer
        return answer
        }

    // Number of comeback attempts possible, before qualifying as worthy
    func attemptsBeforeWorthy() -> BigInt
        {
        if let answer = attemptsBeforeWorthyMatrix[self.us,self.them] { return answer }
        if self.isGameOver { return 0 } // Never qualified for a comeback
        if self.isWorthy
            {
            // Once "worthy", all games from this score are comeback attempts
            return self.gameCount()
            }
        let answer = self.afterWin.attemptsBeforeWorthy()
                   + self.afterLoss.attemptsBeforeWorthy()
        attemptsBeforeWorthyMatrix[self.us,self.them] = answer
        return answer
        }
```

```
    // Number of comeback successes possible, before qualifying as worthy
    func comebacksBeforeWorthy() -> BigInt
        {
        if let answer = comebacksBeforeWorthyMatrix[self.us,self.them] { return answer }
        if self.isGameOver { return 0 }
        if self.isWorthy
            {
            // Once "worthy", all wins from this score are comeback successes
            return self.winCount()
            }
        let answer = self.afterWin.comebacksBeforeWorthy()
                   + self.afterLoss.comebacksBeforeWorthy()
        comebacksBeforeWorthyMatrix[self.us,self.them] = answer
        return answer
        }
    }

let startScore = GameScore( us:0, them:0 )
print( "All games won      =", startScore.winCount() )
print( "Comeback attempts  =", startScore.attemptsBeforeWorthy() )
print( "Comeback successes = ", startScore.comebacksBeforeWorthy() )
```

```
All games won      = 126765060022822940149670320537 6
Comeback attempts  = 130394092938053427729040656421 6
Comeback successes =  100950946076334560732219489624
```