A Study and Simulation of Gambing Laws in Australia

By Lachlan Knell

Introduction

Gambling is often seen as a form of entertainment, with the possiblity of “winning it big”, However the games located at the casino are always designed to have a mathematical advantage for the gambling venue, or house. The goal of this report is too examine and create a ‘game of science’ that complies with casino gambling laws in Australia. The report is designed to use various features of combinatorics to calculate the theoretical probabilities for each of the divisions. A script was also written in C in order to simulate the most amount of games possible to get more accurate results, both of the data sets were then stored in microsfoft Excel. In order to reduce the amount of variables in the exprimental probabilities all of the dice were assumed to be fair. It was also observed that the simulations can help to identify potential issues in the theoretical probabilities.

Results

There were 3 different probabilities that were required to be calculated:

1. 3 of a kind probability

- Counted outcomes manually, 111, 222, 333, 444, 555, 666, 777, 888

$$P = \frac{\sum Outcomes}{Total}$$

$$P = \frac{8}{8^3}$$

$$P = 0.156$$

2. 3 in a row probability

This was a bit more complicated, but after counting all the different digits and then getting their total arrangements EG:

123(3!), 234(3!), 345(3!), 456(3!), 567(3!), 678(3!)

$3! = 6$

$$P = \frac{\sum Outcomes}{Total}$$

$$P = \frac{6 * 6}{512}$$

$$P = 0.07$$

3. Total number on dice is > < probability

Same working for all of the rest of the probabilities, get arrangements and count all different digits

885 (3), 884(3), 876(3!), 875(3!), 866(3), 777, 776(3)

$$P = \frac{\sum Outcomes}{Total}$$

$$P = \frac{(3 * 4 + 3! * 2)}{512}$$

$$P = 0.035$$

Table 1: Calculated Theoretical Probabilities

Divisions	Probability	Prize	Return to Casino		
24	0.001953125	15450	-30.17578125		
3 of a kind	0.015625	6200	-96.875		
>=22 <24	0.017578125	5850	-102.83203125	House edge	15
>=20 <22	0.048828125	4500	-219.7265625		
3 in a row	0.0703125	3500	-246.09375		
Casino	0.845703125	-1000	845.703125		
Payment					
1000					

This Table shows a house edge of exactly 15, a perfect house edge for a casino, it has an extremely rewarding prize of $15450 for the top probability, along with significant prizes for each of the devisions below

Table 2: Original Theoretical Probabilities

Old					
Divisions	Probability	Prize	Return to Casino		
>=24	0.001953125	1000	-1.953125		
3 of a kind	0.015625	750	-11.71875		
>20 <24	0.06640625	650	-43.1640625	House Edge	-91.796875
3 in a row	0.0703125	700	-49.21875		
>18 <20	0.09375	650	-60.9375		
Casino	0.751953125	-100	75.1953125		
Payment					
100					

This table shows a -91.8 house edge, horrible profits for a casino, however the rewards are awesome for the players, which isnt great for a casino.

Table 3: Experimental Probabilities

Experimental Probabilities						
Divisions	Wins	Percentage	Money Made / Run			
			1254738346 (total profit for			
Total Runs	10000000	100.00%	Casino)			
24	19412	0.19%	-29.99154			
3 of a kind	155872	1.56%	-96.64064			
>=22 <24	176039	1.76%	-102.982815	House edge	14.9776255	
>=20 <22	488501	4.89%	-219.82545			
3 in a row	704002	7.04%	-246.4007			
Casino	8456174	84.56%	845.6174			

The experimental results are nigh on identical to theoretical results, proving the probabilities of the theoretical results to be accurate, with a 14.98 house edge compared to a 15.

Evaluation

The theoretical calculations are reasonable because they clearly show a 10-15% house edge for the casino, the rewards that the players get are rewarding, they also show an extremely similar result to the experimental probabilities simulated, proving that the calculated probabilities were accurate. The divisions were orginally designed to imitate a variation of poker with only 3 dice, unfortunately it was found that the dice were unable to simulate the intricacies of poker, instead it was decided to simplify some of the combinations, but the less complicated divisons were kept. As seen in *Table 2,* the divisons were changed in order to reduce the odds of achieving some of them over others. Along with the prizes that were decide on as seen in *Table 1*.

Appendix

Divisions	Probability	Prize	Return to Casino		
24	0.001953125	15450	-30.17578125		
3 of a kind	0.015625	6200	-96.875		
>=22 <24	0.017578125	5850	-102.83203125	House edge	15
>=20 <22	0.048828125	4500	-219.7265625		
3 in a row	0.0703125	3500	-246.09375		
Casino	0.845703125	-1000	845.703125		
Payment					
1000					

Old

Divisions	Probability	Prize	Return to Casino		
>=24	0.001953125	10000	-19.53125		
3 of a kind	0.015625	7500	-117.1875		
>20 <24	0.06640625	6500	-431.640625		-91.796875
3 in a row	0.0703125	7000	-492.1875		
>18 <20	0.09375	6500	-609.375		
Casino	0.751953125	-1000	751.953125		
Payment					
1000					

Experimental	Runs	Percentage	Money Made / Run		
Total Runs	10000000	100.00%	1254738346 (total profit for Casino)		
24	19412	0.19%	-29.99154		
3 of a kind	155872	1.56%	-96.64064		
>=22 <24	176039	1.76%	-102.982815	House edge	14.9776255
>=20 <22	488501	4.89%	-219.82545		
3 in a row	704002	7.04%	-246.4007		
Casino	8456174	84.56%	845.6174		

Script to simulate Games

```
1  #include <stdlib.h>
2  #include <time.h>
3  #include <stdio.h>
4
5  int main()
6  {
7          int num, i, d1, d2, d3, pwin=0, money=0, pwin3fk=0, pwin3row=0, pwin18=0, pwin20=0, pwin24=0;
8          double per=0.0, cas=100.0, hedge;
9          time_t t1;
12         printf("How many times will you be simulating this specific dice rolling simulation?\n");
13         scanf(" %d", &num);
15         srand((unsigned) time (&t1));
16         printf("\n");
17         for (i=0;i<num;i++)
18         {
19                 money-=1000;
20                 /* randomise dice*/
21                 d1=rand() % 8;
23                 d2=rand() % 8;
25                 d3=rand() % 8;
27                 d1++;d2++;d3++;
28                 int dt=d1+d2+d3;
29                 /* compare dice */
30                 if (dt == 24)
31                 {
32                         pwin++;
33                         pwin24++;
34                         money += 15450;
35                 }
38                 if (d1 == d2 && d1 == d3)
39                 {
40                         pwin++;
41                         pwin3fk++;
42                         money += 6200;
43                 }
45                 if (dt >=22 && dt < 24)
```

```
46                  {
47                          pwin++;
48                          pwin20++;
49                          money += 5850;
50                  }
52                  if (dt >= 20 && dt < 22)
53                  {
54                          pwin++;
55                          pwin18++;
56                          money += 4500;
57                  }
59                  if ((d1+1 == d2 && d1+2 == d3) || (d1+1 == d3 && d1+2 == d2) || (d2+1 == d3 && d2+2 == d1) || (d2+1 == d1
&& d2+2 == d3) || (d3+1 == d2 && d3+2 == d1) || (d3+1 == d1 && d3+2 ==d2))
60                  {
61                          pwin++;
62                          pwin3row++;
63                          money += 3500;
64                  }
66          }
68          printf("Player Wins-%d\n", pwin);
69          printf("Money Made-%d\n\n", money);
70          printf("%d-3 of a kind", pwin3fk);
72          per=((double)pwin3fk/(double)num)*100;
73          printf("\n%f\%\n\n", per);
74          cas=cas-per;
76          printf("%d -3 in a row", pwin3row);
77          per=((double)pwin3row/(double)num)*100;
78          printf("\n%f\%\n\n", per);
79          cas=cas-per;
81          printf("%d -20", pwin18);
82          per=((double)pwin18/(double)num)*100;
83          printf("\n%f\%\n\n", per);
84          cas=cas-per;
86          printf("%d -22", pwin20);
87          per=((double)pwin20/(double)num)*100;
88          printf("\n%f\%\n\n", per);
89          cas=cas-per;
```

```
91          printf("%d -24", pwin24);
92          per=((double)pwin24/(double)num)*100;
93          printf("\n%f\%\n\n", per);
94          cas=cas-per;
95
96          printf("%f\n", hedge);
97          printf("%f", cas);
98          return 0;
99  }
```