A Study and Simulation of Gambling Laws in Australia

By Lachlan Knell

Introduction

Gambling is often seen as a form of entertainment, with the possibility of “winning it big”. However the games located at casinos 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, with different prizes in order to achieve a reasonable ‘house edge’.

It was observed in the report that a legal house edge was between 10-15%, which means that the gambling venue should take roughly that percentage of each dollar the players put in, for example if a player plays a game for $10 than the casino should take $1-$1.50 per played game. In addition, it is shown that in most games the lower the probability of an event occurring, the higher the prizes are shown to be. This is normally used to create a sense of fairness to the games, even if the prizes do not match the probability of the event occurring, for example, if a player does the same as before and pays $10 for a game and the winning outcomes have a 10%, 5% and 2% chance of winning, the mathematically fair way of paying the player should be $100, $200 and $500 but in reality the player will often receive something more akin to $20, $50, $100 in order to increases the house edge for the casino.

The report assumes that the dice being used are fair and even, because if the dice are uneven the probability of each event of occurring is unknown it forbids the reasonable examination of the probability of events.

The report also used a variety of mathematical techniques in order to achieve reasonable results. These techniques were: Probability, to calculate the probability of the individual divisions occurring,   The multiplication and addition rule to accurately calculate theoretical probabilities for each of the divisions.

Results

There were three distinct probabilities that were required to be calculated:

1. 3 of a kind probability

This was simple because it only required counting 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 slightly more complicated because it required the use of the multiplication rule to calculate the number of possible outcomes of the numbers, in addition to the arrangements of the individual sets of numbers:

For example 3 different numbers have 3! arrangements, which is 6, allowing the outcomes to be 6*(8, total number of possible outcomes per dice, -2, total dice rolled – 1), which was also 6.

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

3! = 6

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

$$P = \frac{6 * (8-2)}{512}$$

P = 0.07

3. Total number on dice is > < probability

10 combinations of numbers add up to 20

- 1+19, 2+18, 3+17, 4+16, 5+15, 6+14, 7+13, 8+12, 9+11, 10+10

if No>8 then /2

- gives 1+9.5+9.5, 2+9+9, 3+8.5+8.5, 4+8+8, 5+7.5+7.5, 6+7+7, 7+6.5+6.5, 8+6+6, 4.5+4.5+5.5+5.5, 5+5+5+5

if set of numbers has a number that is not whole & if you can round the decimal down and the other up and both are still > 8 then remove *and there is only 3 numbers

- 2+9+9, 4+8+8, 6+7+7, 8+6+6, 5+7+8

if numbers are still greater than 8 remove

- 4+8+8, 6+7+7, 8+6+6, 5+7+8 – there are 4 valid combinations that have a sum of 20

This required use of simple division and addition combined with pre-set rules to calculate the combinations of numbers to get the sum of another number within the constraints of the three sided dice. However the use of the addition and multiplication rule was required, along with knowledge of arrangements to get the total number of outcomes.

488, 677, 866, 875
- If they have two of the same digit than the possible arrangements is 3, if they have 3 different numbers than it is 3!

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

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

$$P = 0.035$$

Table 1: 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 presents an extremely low house edge, with the casino losing money on each hand, however if features rewarding prizes for the players, incentivise players to return to the game. However the table shows unreasonable results because the house edge is below the 10-15% observed amount required for a casino.

Table 2: 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 15, within the legal limits and still making the casino money. It features an extremely rewarding prize of $15450 for the top prize, also incentivising players to return to play, however this time the players are losing money in the long term instead of gaining money. This in turn presents that the results here are reasonable because of the house edge and relevant prizes

Table 3: Experimental Probabilities

Experimental Probabilities				$1254738346 (total profit for Casino) ($)	
Divisions	Wins	Percentage	Money Made / Run		
Total Runs	10000000	100.00%	Return to 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		

```
for (i=0;i<num;i++)  /* For is a loop, basically saying "Run this for Num amount of times" */
        {
                money-=1000; /* Subtract betting money */
                /* randomise each of the dice*/
                d1=rand() % 8;
                d2=rand() % 8;
                d3=rand() % 8;
                /* Account for random function doing 0-7 instead of 1-8 */
                d1++;d2++;d3++;
                /*Calculate the total of all the dice rolled to allow for easier calculations later on in the script*/
                int dt=d1+d2+d3;
```

Text 1: Snapshot of Programming for simulated dice rolls

The Experimental results coincide with the theoretical calculated results. They both show very similar house edges and Casino returns, communicating that the results are reasonable. It was chosen to do 10,000,000 trials because it was quick to run and allowed for exceptionally accurate and reasonable results.

It was chosen to simulate the games via a script because it enabled more efficient, quicker and more flexible trials than using other techniques such as excel. It also allowed the simulations to be run far more times in a less labour intensive manner than via other methods.

Discussion

The results achieved were reasonable because they clearly follow the observed 10-15% house edge required for a casino in Australia, The rewards also follow the observed logical ascending nature of prizes received by the players in the casino. The assumptions made about the probabilities of both theoretical and experimental probabilities increase the reliability and reasonability of the results gathered and calculated this is because both the theoretical and experimental results assume that the outcomes are equally likely, enabling the simulations and calculations to be comparable, instead of having different possible outcomes.

The ability of the simulation to generate exceptionally large numbers of trials in seconds allows it to more accurately model the game over extremely large stretches of time, allowing for potential discrepancies in the calculations and simulations to appear because the simulations are extremely reliable. In addition the use of Excel to calculate the house edge and probabilities allowed for extremely fast and flexible calculations, it also allowed for quick changes and records to be kept of these changes.

However, the report had a few limitations. The simulations were unable to be quickly implemented into excel, requiring manual input instead of automated changes, which slowed the process of examining the simulated data significantly. However it would be possible to alleviate this problem by directly editing the excel file with the script, unfortunately that is out of the scope of this report and would require a level of knowledge unavailable within the time allocated to simulate the games.

Conclusion

In conclusion the report was examining the mathematics that casinos use to extort profits out of their customers, along with the laws that constrain the casinos predatory behaviours. It was discovered that it is possible to create a house edge that complies with the laws of gambling in Australia, it also sheds light on the amount of profits casinos can make with an apparently small house edge.

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

```
#include <stdlib.h>
#include <time.h>
#include <stdio.h>

int main()
{
    int num, i, d1, d2, d3, pwin=0, money=0, pwin3fk=0, pwin3row=0, pwin18=0, pwin20=0, pwin24=0;
    double per=0.0, cas=100.0, hedge;
    time_t t1;
    printf("How many times will you be simulating this specific dice rolling simulation?\n");
    scanf(" %d", &num);
    srand((unsigned) time (&t1));
    printf("\n");
    for (i=0;i<num;i++)
    {
        money-=1000;
        /* randomise dice*/
        d1=rand() % 8;
        d2=rand() % 8;
        d3=rand() % 8;
        d1++;d2++;d3++;
        int dt=d1+d2+d3;
        /* compare dice */
        if (dt == 24)
        {
            pwin++;
            pwin24++;
            money += 15450;
        }
        if (d1 == d2 && d1 == d3)
        {
            pwin++;
            pwin3fk++;
            money += 6200;
        }
```

```
		if (dt >=22 && dt < 24)
		{
			pwin++;
			pwin20++;
			money += 5850;
		}
		if (dt >= 20 && dt < 22)
		{
			pwin++;
			pwin18++;
			money += 4500;
		}
		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))
		{
			pwin++;
			pwin3row++;
			money += 3500;
		}
	}
	printf("Player Wins-%d\n", pwin);
	printf("Money Made-%d\n\n", money);
	printf("%d-3 of a kind", pwin3fk);
	per=((double)pwin3fk/(double)num)*100;
	printf("\n%f\%\n\n", per);
	cas=cas-per;
	printf("%d -3 in a row", pwin3row);
	per=((double)pwin3row/(double)num)*100;
	printf("\n%f\%\n\n", per);
	cas=cas-per;
	printf("%d -20", pwin18);
	per=((double)pwin18/(double)num)*100;
	printf("\n%f\%\n\n", per);
	cas=cas-per;
	printf("%d -22", pwin20);
	per=((double)pwin20/(double)num)*100;
	printf("\n%f\%\n\n", per);
```

```
    cas=cas-per;
    printf("%d -24", pwin24);
    per=((double)pwin24/(double)num)*100;
    printf("\n%f\%\n\n", per);
    cas=cas-per;

    printf("%f\n", hedge);
    printf("%f", cas);
    return 0;
}
```