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, with different prizes in order to achieve a resonable ‘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 about that percent of what 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 occuring, 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 occuring, 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 occuring 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 occuring, 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

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

P =

P =

P =

2. 3 in a row probability

This was slightly more complicated because it required the use of the multiplication rule to calculate the combinations of the numbers to get the different arrangements of the numbers. For example 3 different numbers have 3! combinations, which is 6, allowing the outcomes to be 6\*Different Numbers, which was also 6.

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

3! = 6

P =

P =

P = 0.07

3. Total number on dice is > < probability

This required the use of both the addition rule and multiplication rule in order to calculate the outcomes for these, for if the numbers had two numbers the same, there was only 3 possible outcomes, but if they were all different numbers the outcomes could be 3!.

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

P =

P =

P =

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, incentivsing players to return to the game

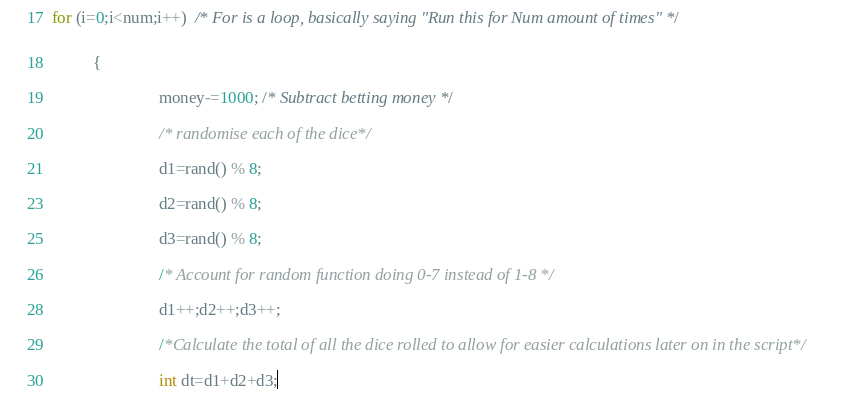
Table 2: Calculated Theoretical Probabilities

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Divisionsf | 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.

Table 3: Experimental Probabilities

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Experimental Probabilities |  |  |  |  |  |
| Divisions | Wins | 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 |  |  |

Text 1: Snapshot of Programming for simulated dice rolls

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 }