The Profitability of Games of Chance

New Mexico Adventures in Supercomputing Challenge
Final Report
April 6th, 2005

Team 58
Santa Fe High School

Team Members
Leif Hopkins

Teacher
Anita Gerlach

Table of Contents

Executive Summary

The “art” of gambling is one which has been pursued for many centuries. Unwilling to join the workforce as citizens chained to the obligation of “jobs,” knaves from past and present become slaves to the science of probability: a single roll of the die, spin of the wheel, or turn of the card determining whether or not they leave in a luxury car and Italian suit, or with trash bags on their feet and no shirt on their back. Indeed, for these individuals, it becomes quite important to understand the “science” behind their financial ups and downs.

This project set out to explore these ups and downs: via simulation, this project has set out to understand the probabilistic nature of these games. Using simple C++ programming techniques, this project used randomly generated numbers as a base for simulating simple games of chance, specifically, dice, craps, and roulette. By modifying the parameters of the game, that is, the wager made, the potential loss and gain, the iterations of the simulation (number of gambles), the nature of these games was explored with very interesting results.

In particular, it became very clear very early that these games require patience and intelligence if a win is desired. Indeed, certain patterns of bets and losses produced certain patterns or profit--indicating some degree of skill in the games. Erratic betting patterns and wagers too large for the gamblers “bankroll” (total funds) proved in ruination, or complete loss, upon nearly every trial.

Final Report

Introduction

This project had its birth in the middle of America’s poker craze. With poker everywhere: television, casino’s, and home’s across the United states, the thought occurred to me--what is the nature of this game? Is it a skillful game, or is it simply a probabilistic game, pure luck, as it were?

This question is certainly a topic of debate, and I will not discuss it here. I will say, though, that this question of “luck versus skill” perplexed me. I wondered if perhaps it was applicable to other gambling games--was it possible to analyze these games using mathematical techniques, deciphering a hidden strategy? Perhaps, via computer simulation, patterns and techniques could be found. Ergo, my project was born.

Upon only a light degree of study, it can be shown that because of the universe’s adherence to the mathematics of probability (from the weather to the quantum interactions of the particles therein), that, indeed, even games of pure chance can be influenced by parameters set forth by the player.

This project may be a numerical verification of these strategies set forth by principles of probability theory. If adherence to these rules denotes maximized profit, then they are true to a great extent. However, if deviation from these rules results in no or little change of the profit potential, then the truth of these strategies is consequently reduced.

However, at its core, this project was really a study of the *nature* of these games. How do wagers, gambles, and other changes to their specifications change the outcome of the gambler? This project set out to answer this question.

Description

These games of chance would be simulated by a computer program using C++ computer code.

Undertaking the project, it become necessary to supplement parameters by which to perform it. The games of chance explored would be limited to 3 simplified games. The first would be a game of dice involving only a single die, wherein the winning throw was simply one side of the dice. The second would be a simplified “craps” game, where two dice were thrown and totaled--a total of 7 or 11 being a win, any other being a loss. Finally, the third game in question was roulette, where the player would bet on only one of the 36 numbers located on a theoretical roulette wheel (normally, they are offered many other options, but for the sake of project simplification, I offered them only this) and be paid 36 to 1 on their wager if they won. The simulation would end if the player became “ruined,” or lost all of their funds.

The probabilistic nature of the games was simulated using a seeded random number generator function. I would specify, for example, the generation of random integers from 1 to 6, which would simulate each side of a die.

The strategies implemented in this game would be modified by the use of an interface built into the program. For example, each program would prompt the user on start-up to enter the specified variables in question. The modification and alteration of these variables would then produce results which could be analyzed.

Results were analyzed via plotting of monetary fluctuation during a gambling session and the ultimate win or loss of the gambler. Differing parameters were analyzed in an effort to provide insight into which ones produced maximum profit.

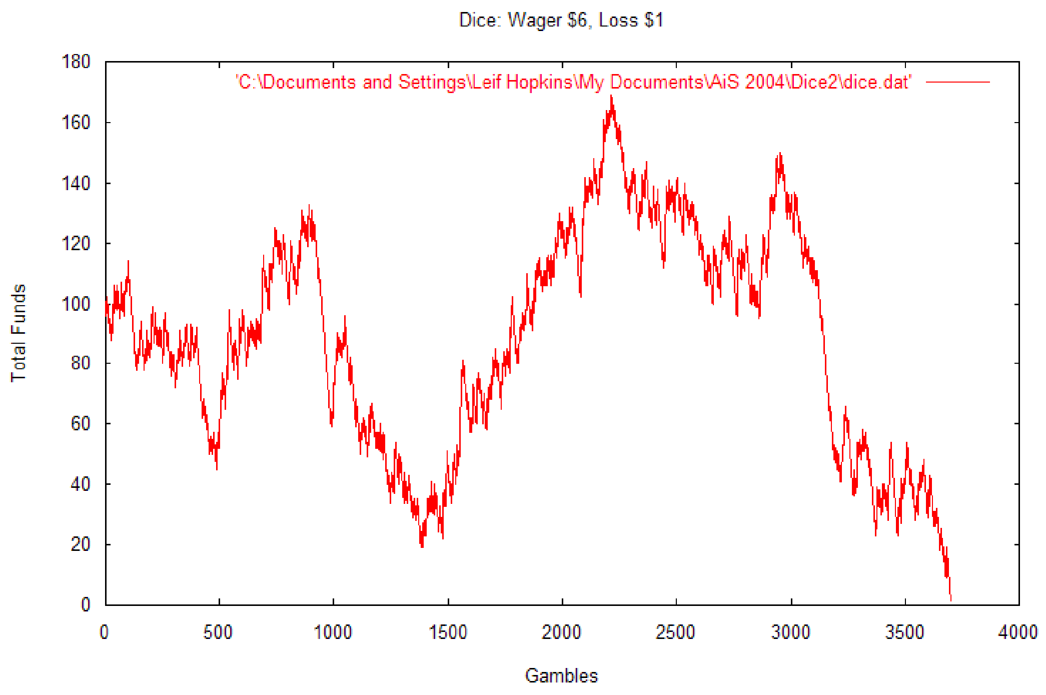
Results

The following results are presented one game at a time.

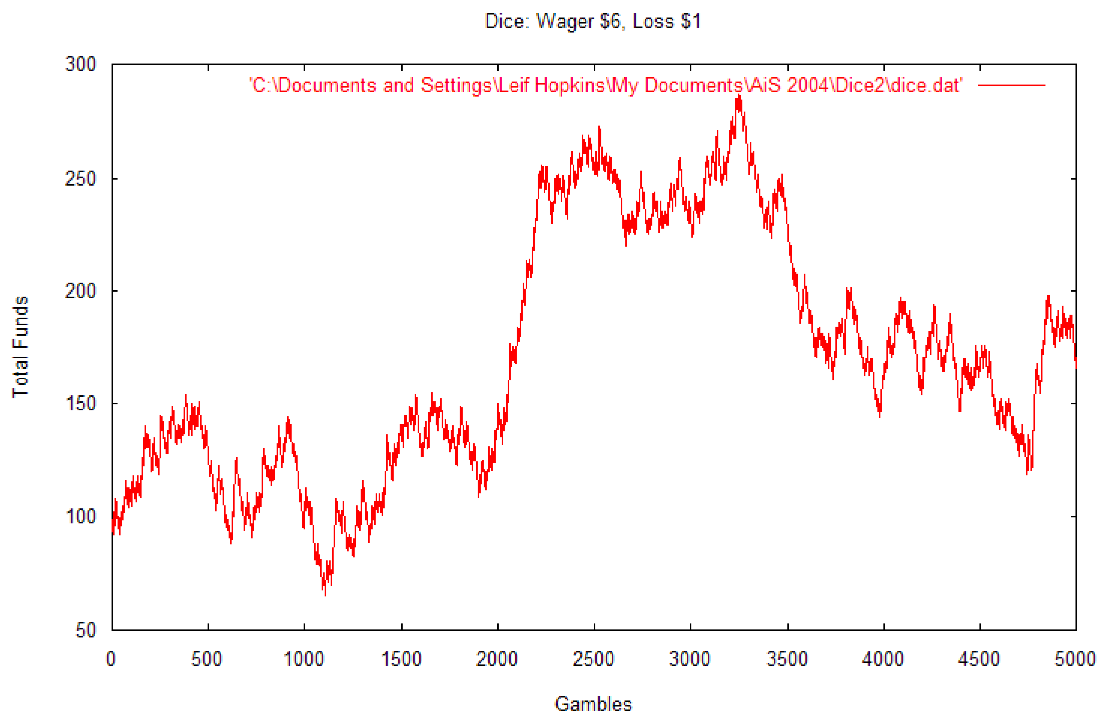
DICE

The dice game analysis took into account the wager, which was added to the gamblers funds if a 1 was thrown, or a loss, which was subtracted if he lost. The gambler starts of with $100 dollars. A gambling session was 5000 throws.

The first result of these experiments found that a wager of $6 with a potential loss of $1 was an equilibrium, producing a profit approximately ½ of the time and a loss the other half.



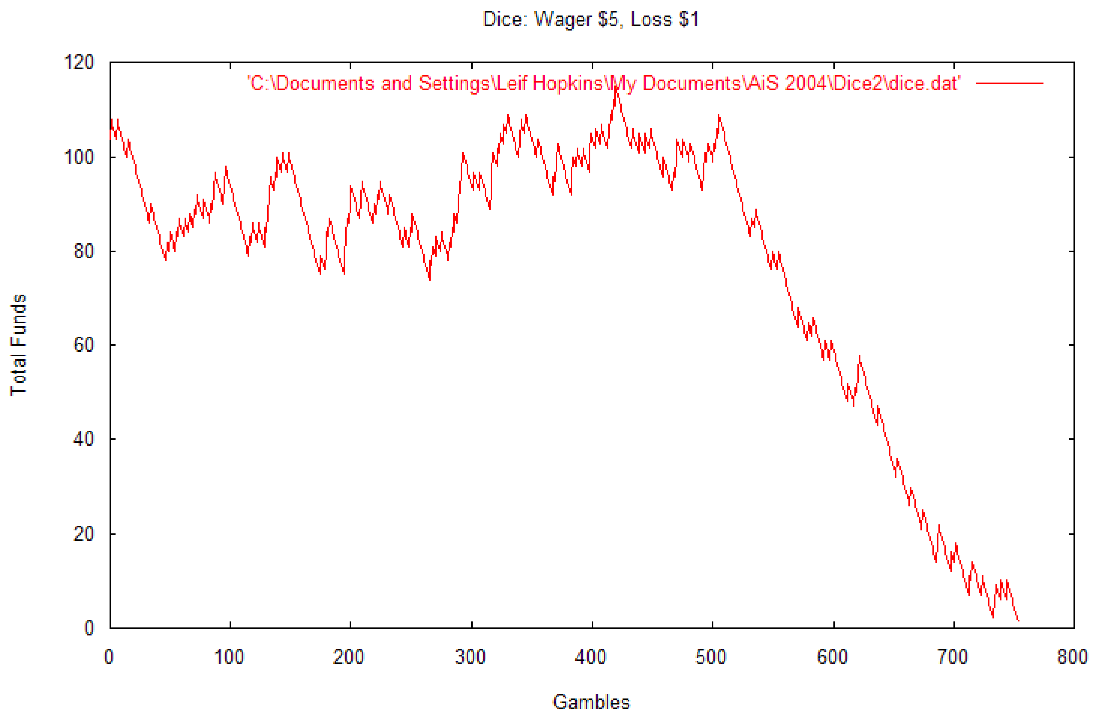
This experiment resulted in ruination.



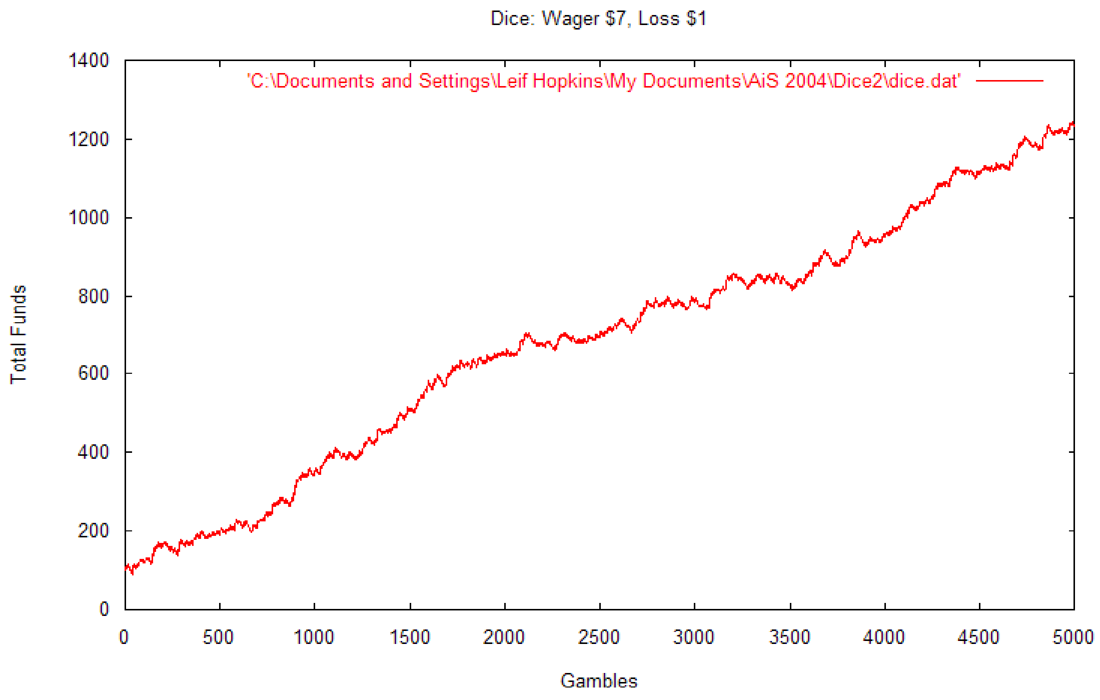
This experiment resulted in a profit of approximately $80 dollars.

These results are very mathematically verifiable; a simple analysis of this game shows that 1/6 of the time, the player should win. Theoretically, then, shouldn't they break even? Not always. The law of averages, a law of probability, states that as we throw a dice, the greater the number of throws, the more likely that, say, a 6 is thrown about 1/6 of the time. However, it does not say that the number of throws of a 6 will equal that of any other throw, in fact, it would be astonishing if it did. We may conclude that a wager equaling the *expectancy* of this game, that is, a loss of $1 and a win of $6 for a ratio of 1/6, or the probability of a win, is the equilibrium of the game which results in ½ chance of attaining profit.

Indeed, way may further conclude that modifications to the bets beyond this ratio, either exceeding or lowering it, will result in ultimate profit or loss *most* of the time. Consider the following graphs.



With a wager of $5, and a potential loss of $1, the expectancy of 1/6 is reduced, making this an unfair game. The gambler is inevitably ruined.



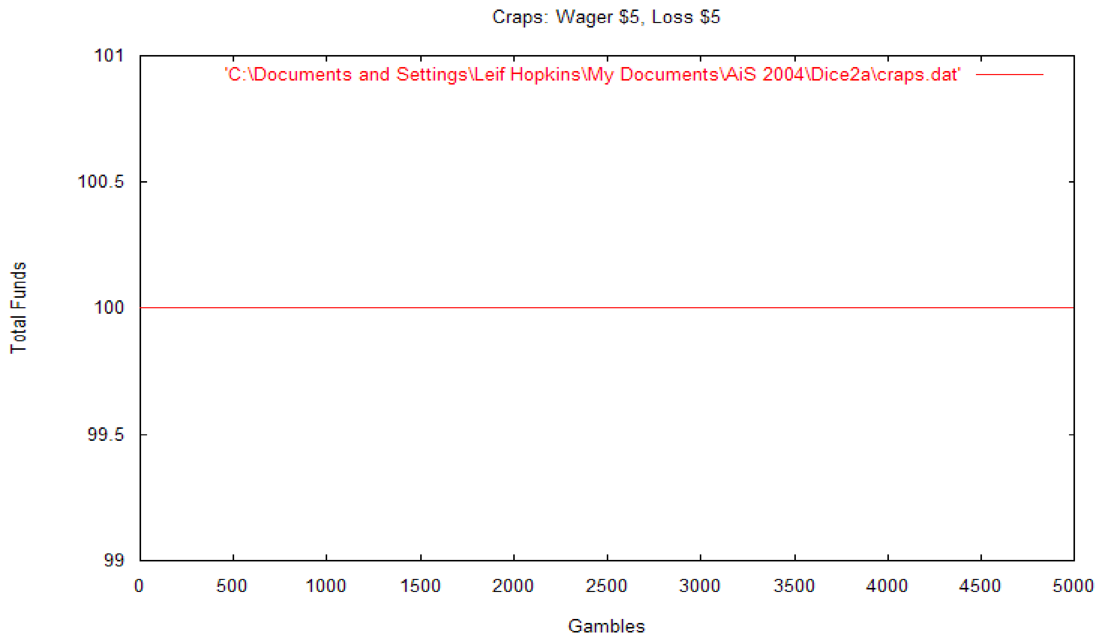
With a wager of $7, and a potential loss of $1, the expectancy of 1/6 is exceeded, making this game unfair *but in the players advantage.* The player ultimately profits.

We are *never* to assume that these results are not to be defied. Probability is uncontrollable--it is possible to lose when the expectancy is exceeded and win when it is reduced--but it is very *unlikely.*

CRAPS

The game of craps was exactly the same in scope as the game of dice, except that two dice were simulated (the total of one dice was determined by a random integer of 1 to 6 and added to the total of the another similarly determined dice to accurately simulate the relationship). The same variables were considered, that is, the wager and the potential loss. On this occasion, though, a total of 7 or 11 is a win, and anything else is a loss. The gambler begins with $100 dollars, a gambling session is 1000 throws.

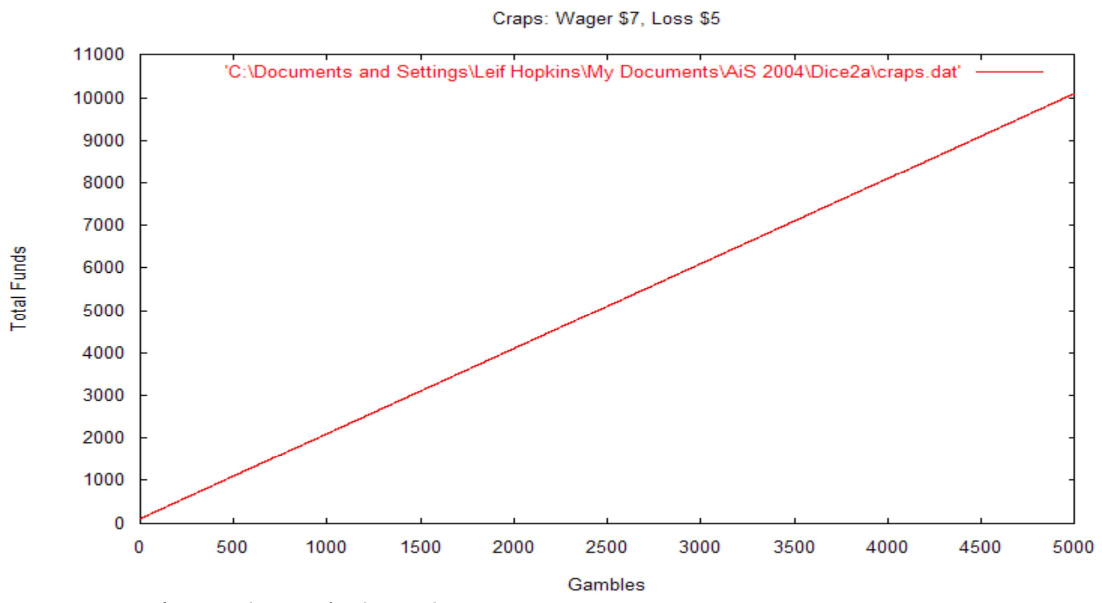
Perhaps the most interesting thing about this game is that the equilibrium, which for the dice game existed at a wager of $6 and a loss of $1, exists when the wager and the loss are equal. Additionally, the equilibrium is *exactly* the size of the funds when the player began. The player breaks perfectly even. Please refer to the graph below.



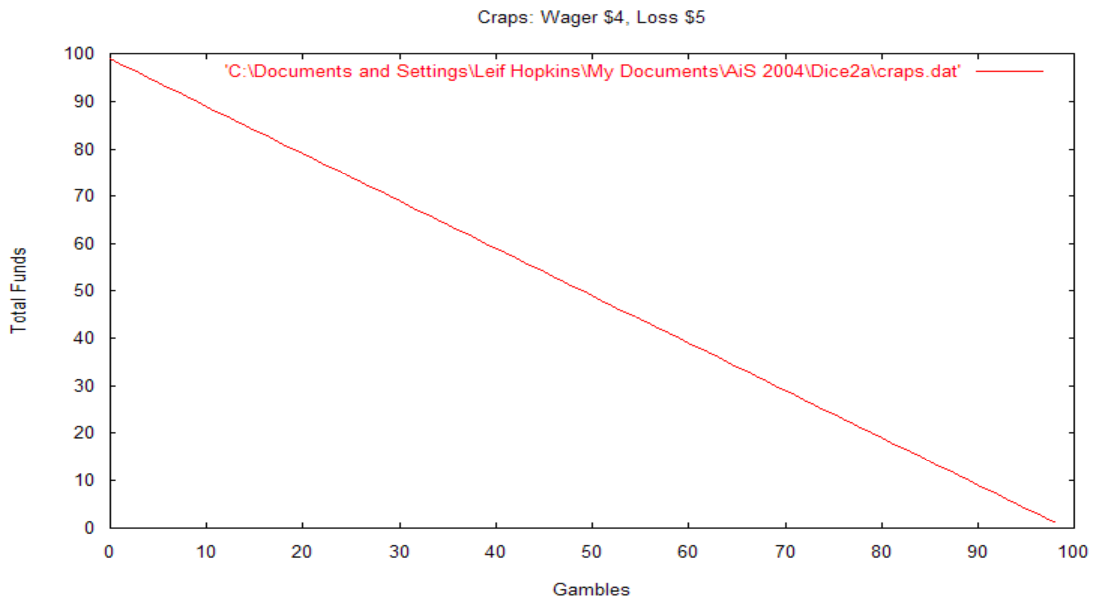
The gambler stays even the entire session.

We may conclude that the above result from the odds of throwing a 7 or 11 in craps, which can be determined to be close to ½ of the time. Therefore, the player breaks exactly even.

Similarly, fluctuation in this 1:1 ratio of loss versus wager will result in profit or loss. Refer to the graphs below.



A 7:5 wager to loss ratio results in a win.



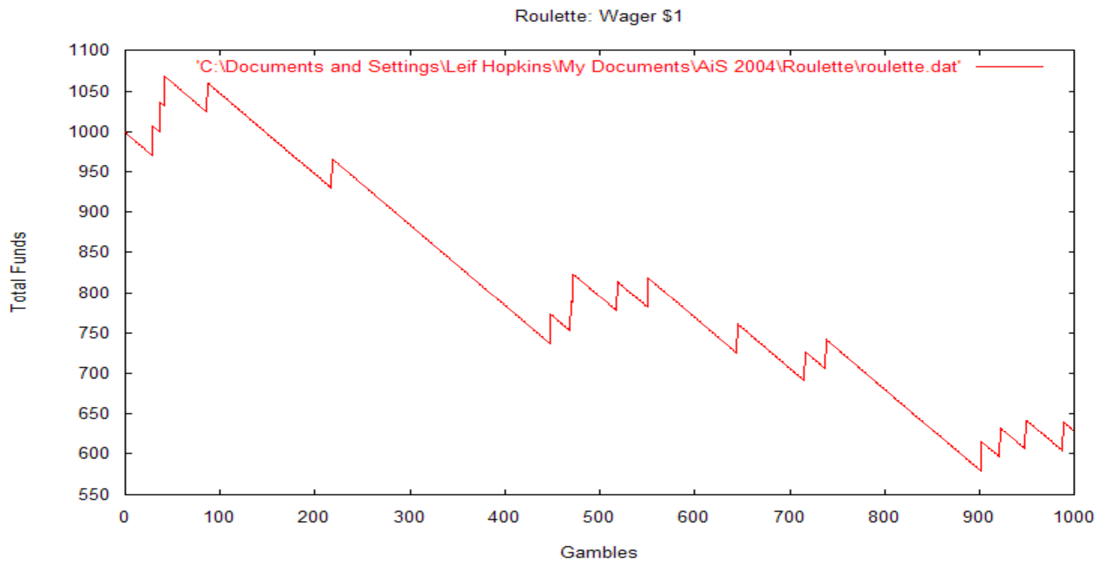
A 4:5 wager to loss ratio results in a loss.

Craps is therefore a much more simplistic game than dice. The ½ probabilities of a win mimic, say, a flipping coin. Any modification to the wager-loss ratio results in a consequent win or ruin, depending on if the ½ expectancy is lowered or exceeded.

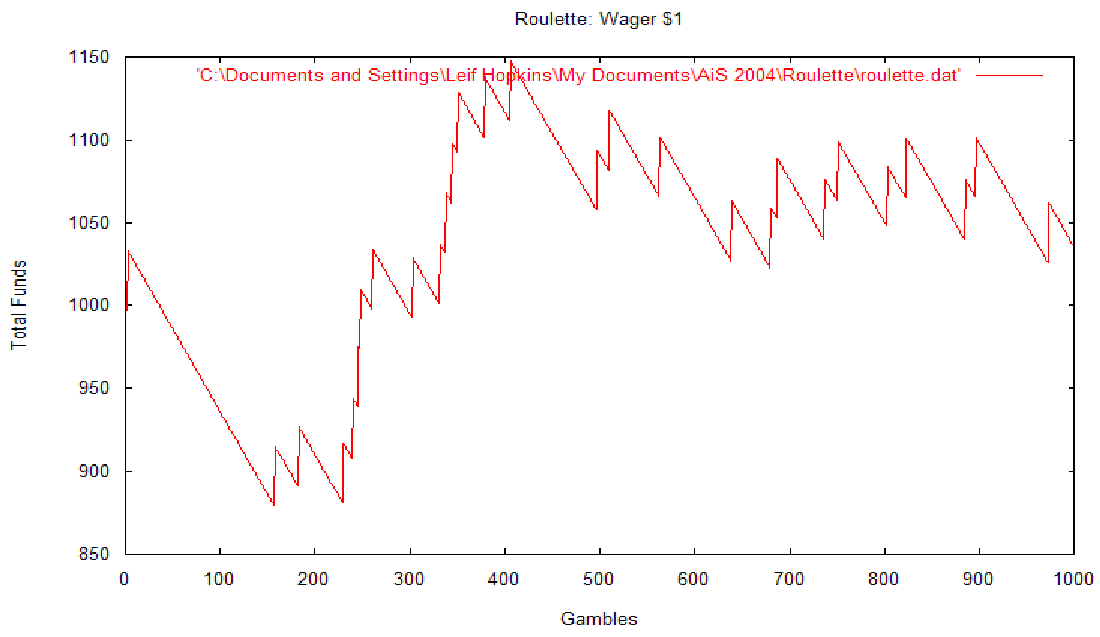
ROULETTE

My roulette simulator worked by randomly generating a number from 0 to 36, mimicking the makeup of a standard roulette wheel and ball. The interface of the program allowed the user to choose a number 0 to 36 (perhaps for mere posterity, as the 1/37 odds were equal regardless) and the wager, which was paid 36 to 1, or multiplied by 37, upon a win. The gambler was given larger funds due to his lowered odds--this time $1000 dollars. A gambling session this time was 1000 spins.

The funds of the gambler during a roulette session were characterized by immense “swings,” or fluctuations in funds--likely the result of his low odds to win, but large pay off. Consequent to the nature of the game, the odds of a win or loss were seemingly quite random. Refer to the following graphs.

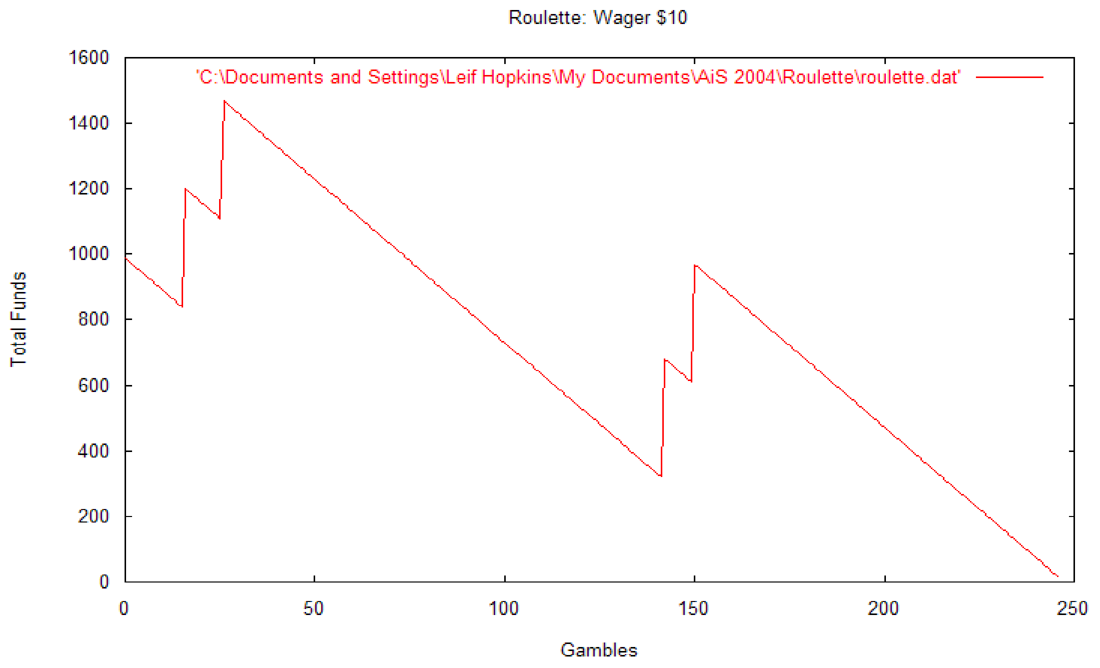


Wagering $1, the gambler ultimately loses money, despite his wins.

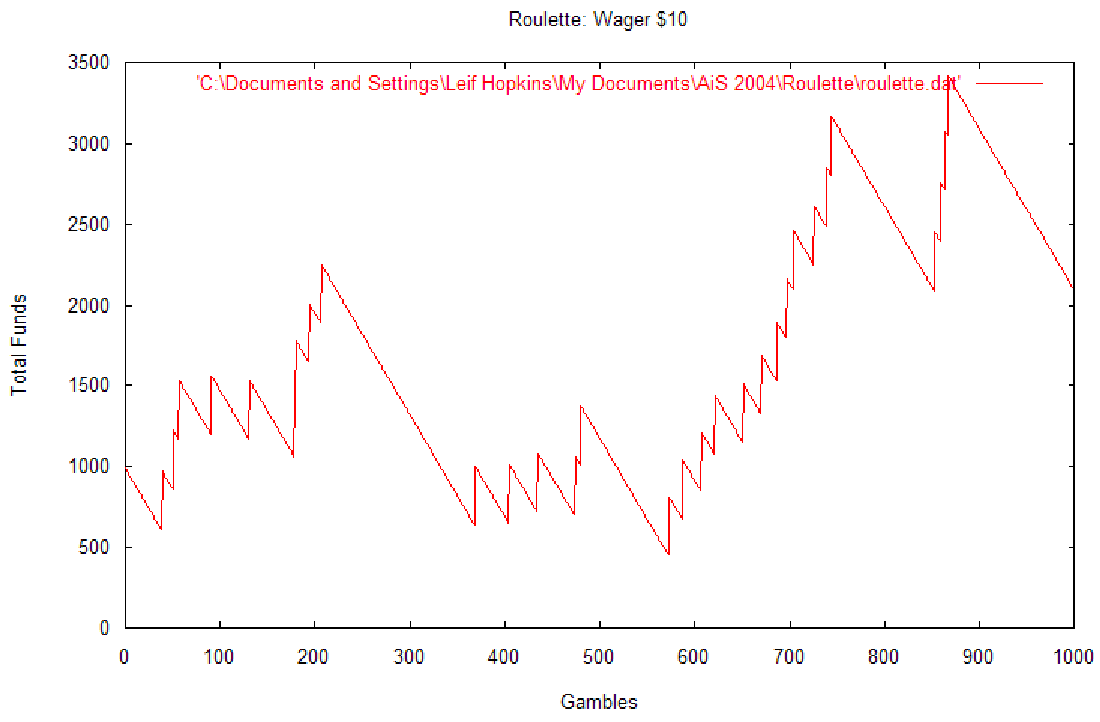


Wagering $1, the gambler ultimately wins money despite his intial and subsequent losses.

The only difference between larger and smaller wagers was the potential pay off. Please refer to the graphs below.

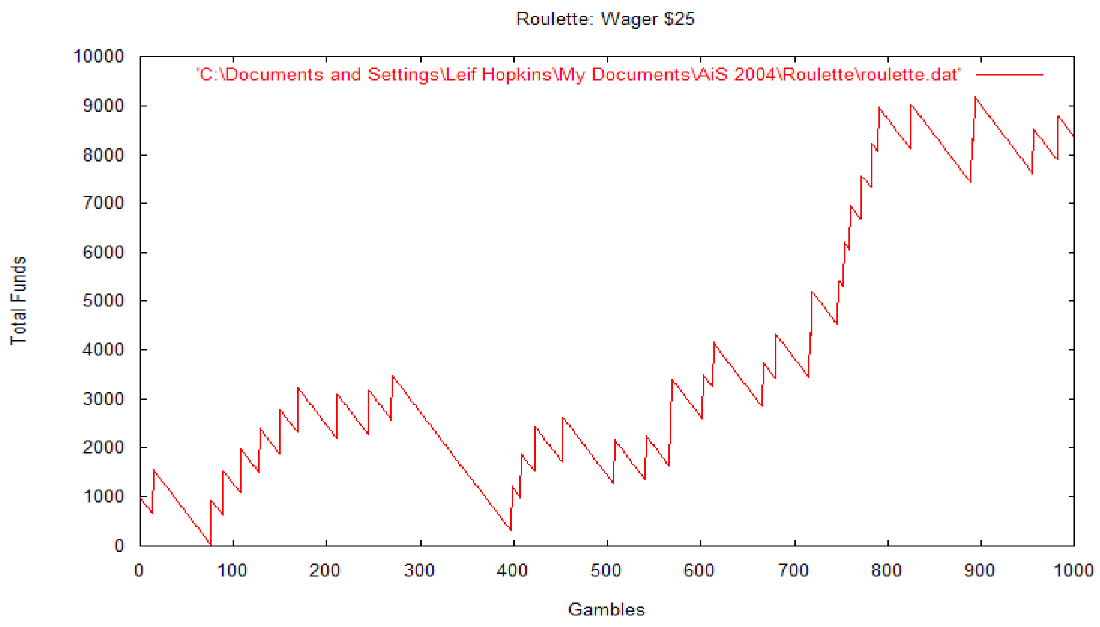


Wagering $10, the gambler is ruined quickly into the session.

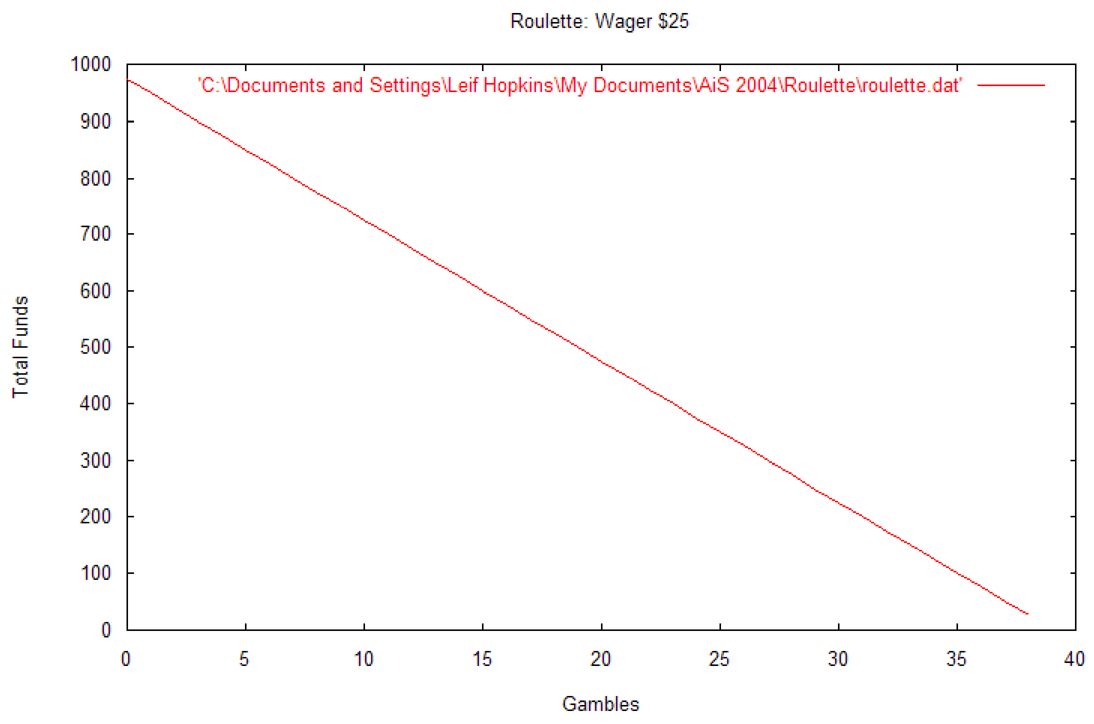


This time, the gambler profits about $1100 dollars, despite his "swings."

Again, repeat the experiment with a wager of $25 per spin. Please refer to the graphs below.



Wagering $25 and almost losing everything, the gambler ultimately profits almost $8500 dollars.



The gambler is ruined almost instantly.

Roulette is obviously a very unpredictable game. The risks, however, are matched by the rewards. It would seem sensible to bet only a small amount, though, unless you are willing to risk everything to be rich.

The odds of a win versus a loss over a session appeared to be somewhere near ½, but that is unverifiable. It did seem that as the wager increased, it took a larger number of experiments to produce a profitable result--the result therin, though, was usually immensely profitable.

Conclusions

I am able to draw two conclusions from this project: (1) there are parameters, which if under player control, will make one profit from a game of chance, and (2) adherence to sound strategies set forth by principles of probability theory result in lowered losses and maximized profit. Take for example the game of dice; if the player was foolish enough to not wager an amout exceeding the expectancy, then he/she would certainly be ruined. In the roulette game, if the player wagered an amount of money in proportionate to their total funds, they were ruined more than if they had wagered a sensible amount.

This project has also made a conclusion about slight mathematical edges over time. Think back to when our dice player wagered lower than his expectancy: remember his ruination, despite only his slight disadvantage? This is the same method exploited by a common casino: just think, they are allowed a 2% edge over the player--not much, right? Wrong.

We can see that the nature of these games can be explained in terms of probability. The numerically determined results we attained can be explained in simple probabilistic terms, some not even requiring any formal education on the subject. Since this analysis is readily available, strategies and techniques can be developed which will assist a player involved in these games of chance.

That leads me to the most significant original achievement of the project: the proof that adherence to strategy can even be applied to games of chance. As long as a parameter is under player control, then he is able to make decisions regarding his situation. Consequently, he may strategize, to a point--even when his fate is not up to him. Perhaps this conclusion can be easily drawn…but can the strategies be derived from mere visceration? Hardly. This project has aptly demonstrated numerically verifiable techniques to maximize and minimize profit in simple games of chance, techniques which the common gambler often ignores abandons for the pursual of a “lucky streak,” or other fallacy. We see from this project that profit is produced by a sound mind, not a “hot hand.”

Recommendations

This project is nowhere near the state of completion at which it could be. Many things can be done to expand upon the notions presented henceforth. For one, the C++ code used to determine the aforementioned results is quite archaic, and can be easily expanded to take into account multitudes of other parameters. Secondly, the games explored in this project were themselves painfully simple--not subject to any other limitations except to be randomly determined. Indeed, this project would take a new turn if it were to be applied to a more complex game, say Blackjack, or even various forms of poker.

Acknowledgements

I would like to thank my teacher sponsor, Anita Gerlach, in her assitance during my project. I would also like to thank the help of several internet resources on my topic, as well as some help with C++ coding techniques. Most of all, I would like to thank the American poker craze, which despite entertaining me immensely, has opened up my eyes to the fascinating mathematics of games, logic, and probability more so than they ever were before (I've always been an analysis fan…it's almost as if I didn't even know math had other branches!).

APPENDICES

C++ Computer code

```
1. Dice program
-----------------------------------------------------------------------
/*

  This program simulates the game of dice
  by using a random number generator to choose
  integers from 1 to 6, corresponding to each
  side of the die.  A 1 is a win.  Anything else
  is a loss.

*/

#include <cstdlib>
#include <iostream>
#include <ctime>
#include <fstream>

using namespace std;



int dice;
int gambler1;
int gamble;
int loss;

void main()
{
      ofstream outData;
      outData.open("dice.dat");
      outData.setf(ios::fixed);



      cout<<"Please enter the gamble you wish to take."<<endl;
```

```
        cin>>gamble;
        cout<<"Please enter the potential loss."<<endl;
        cin>>loss;

        gambler1 = 100;

        srand(static_cast<unsigned>(time(0)));

        for (int index=0; index <5000; index++)
        {
              dice = 1 + (rand ()%6);

              if (dice == 1)
              {
                    gambler1 = gambler1+gamble;
              }

              if (dice == 2 || 3 || 4 || 5 || 6)
              {
                    gambler1 = gambler1 - loss;
              }

              if (gambler1 <=0)
              {
                    cout<<"You are ruined."<<endl;
                    break;
              }

              outData<<gambler1<<"\t"<<endl;


              cout<<dice<<endl<<endl;

        }

        cout<<gambler1<<endl;
  }
```

----

2. Craps program

```
#include <cstdlib>
#include <iostream>
#include <ctime>
#include <fstream>

using namespace std;

int dice1;
int dice2;
int total;
int gambler1;
int gamble;
int loss;

void main()
{

      ofstream outData;
      outData.open("craps.dat");
      outData.setf(ios::fixed);

      cout<<"Please enter the gamble you wish to take."<<endl;
      cin>>gamble;
      cout<<"Please enter the potential loss."<<endl;
      cin>>loss;

      gambler1 = 100;

      srand(static_cast<unsigned>(time(0)));

      for (int index=0; index <5000; index++)
      {
            dice1 = 1 + (rand ()%6);
            dice2 = 1 + (rand ()%6);
            total = dice1 + dice2;

            if (total == 7 || 11)
            {
                  gambler1 = gambler1+gamble;
            }



            if (total != 7 || 11)
            {
                  gambler1 = gambler1 - loss;
            }

            if (gambler1 <=0)
            {
                  cout<<"You are ruined."<<endl;
                  break;
            }

            outData<<gambler1<<"\t"<<endl;



            cout<<total<<endl<<endl;

      }

      cout<<gambler1<<endl;
}
```

----
3. Roulette Game

```
#include <iostream>
#include <cstdlib>
#include <ctime>
#include <fstream>

using namespace std;
```

```
int roulette;
int gambler1;
int wager;
int number;

void main()
{

      ofstream outData;
      outData.open("roulette.dat");
      outData.setf(ios::fixed);

      cout<<"Enter a number from 0 to 36."<<endl;
      cin>>number;
      cout<<"Enter a wager."<<endl;
      cin>>wager;

      gambler1 = 1000;

      srand(static_cast<unsigned>(time(0)));

      for (int index=0; index<1000; index++)
      {
            roulette = (rand () % 37);

            if (roulette == number)
            {
                  gambler1 = gambler1 + (wager * 36);

            }

            if (roulette != number)
            {
                  gambler1 = gambler1 - wager;

            }
            if (gambler1 <=0)
            {
                  cout<<"You are ruined."<<endl;
                  break;
            }


            outData<<gambler1<<"\t"<<endl;


            cout<<roulette<<endl<<endl;

      }

      cout<<gambler1<<endl;
}
```

References

[1] Adams, Joel et al, “C++, An Introduction to Computing,” Upper Saddle River, NJ 07458. 1998, Pretice-Hall, Inc. Second Edition.

[2] Fitzpatrick, Richard, “What is probability?”, The University of Texas, May 2002. http://www.farside.ph.utexas.edu/teaching/sm1/leactures. Accessed 2-3-2005.

[3] Jacobs, Robert, “C++ Tutorials - C++ Random Numbers,” copyright 2003-2004. http://www.robertjacobs.fsnet.co.uk/random.htm. Accessed 4-2-2005.

[4] Webb, Peter, “Probability Theory - The Laymans guide to probability,” 2005. http://www.peterwebb.co.uk/probability.htm. Accessed 2-3-2005.

Note: All graphs were made in Microsoft GnuPlot.
C++ Compiling was performed in Microsoft Visual C++.