CALIFORNIA STATE SCIENCE FAIR
2007 PROJECT SUMMARY

Name(s)
Keegan M. Mann

Project Number
J1309

Project Title
Pseudo-Random Numbers

Abstract

Objectives/Goals
I analyzed the quality of several random number generation algorithms and tested which constants performed best in some of the algorithms. The quality of pseudo-random numbers is crucial to the accuracy of a Monte Carlo simulation and the security of encrypted documents.

Methods/Materials
In my experiment I generated and tested sequences from different algorithms with a program I created in Visual Basic. To evaluate the randomness, I calculated pi with the Monte Carlo method, found the length of the period until the sequence repeats, and I calculated the average. I also measured the effect of changing the constants of the algorithm on the output sequence.

Results
Blum Blum Shubwas best overall because it was best on the most important tests although it was fifth on the test of the average. I was wrong in my prediction that it would be best at all of the tests.
The several versions of the lagged Fibonacci generator performed differently. The exclusive-or version did worst on all of the tests. The multiplication version had repetition but did well on the Monte Carlo pi test and the mean test. The addition version had no repetition, did tolerable on the Monte Carlo pi test, and did well on the mean test. The subtraction version did satisfactory on all the tests although it was not the best.
The Middle square method is not very good, because it repeats, sometimes very quickly. The Linear Congruential algorithm is very bad because although it calculated close to pi and doesn't repeat quickly, when points are graphed from the random numbers, they create a non-random pattern.
My hypothesis was wrong in that the repetition did not always get consistently longer with larger values for M in the Blum Blum Shub formula. For powers of ten, the repetition period was five times larger than the previous power. I also found out that prime numbers seemed to do better than non-prime numbers even when they were close in value.
In the Middle Square formula, contrary to my hypothesis, the repetition length was not directly proportional to the number of digits in the output.

Conclusions/Discussion
Different tests gave different algorithms the varying ratings. I believe that the most important tests were the repetition test and the Monte Carlo value of pi test.

Summary Statement
I analyzed the quality of several random number generation algorithms and tested which constants performed best in some of the algorithms.

Help Received
I received no help