Nolan Aubuchon

CS 430

Dr. Wahjudi

March 12, 2017

Random Number Generator Design

For this project, I used the Blum-Blum-Shub Algorithm (Indiana University Bloomington, 2001)

According to Indiana University in Bloomington (para. 3), the Algorithm is as follows:

…where *M* is the product of two prime numbers (para. 4). It further states that the random number generator will only output the parity of *xn+1* (para. 4)

In my implementation of this algorithm, I have selected the prime numbers 127 and 223. Using a seed of 25, here are the first 10 numbers the algorithm generates:

|  |  |
| --- | --- |
| Xn+1 | Output |
| 25 | 1 |
| 625 | 1 |
| 22452 | 0 |
| 6825 | 1 |
| 20901 | 1 |
| 376 | 0 |
| 28092 | 0 |
| 24120 | 0 |
| 4418 | 0 |
| 5555 | 1 |

A more in-depth discussion of the results will be talked about in the results paper.

Why is this algorithm hard to predict? It is the case because only the values of zero or one are shown to the user, i.e. only the computer knows the values of Xn+1. This makes Xn+1 impossible to reconstruct, which is similar to how hash functions work.

# Bibliography

Indiana University Bloomington. (2001, May 30). *Blum Blum Shub Generator*. Retrieved March 12, 2017, from Indiana University Bloomington: https://www.cs.indiana.edu/~kapadia/project2/node11.html