كلية الحاسبات والذكاء الإصطناعي

SC311
Modeling and Simulation

Lecture 07

Dr. Ahmed Hagag

Faculty of Computers and Artificial Intelligence
Benha University

Spring 2023

Random-Number Generation

Chapter 5: Ran. Num. Gen.

- Properties of Random Numbers.
- Generation of Pseudo-Random Numbers.
- Techniques for Generating Random Numbers.
- Tests for Random Numbers.

Properties of Ran. Num. (1/3)

Introduction (1/3)

- Random numbers are central in applications such as simulations, electronic games (e.g. for procedural generation), and cryptography.
- A simulation of any system or process in which there are inherently random components requires a method of generating or obtaining numbers that are random, in some sense. For example, the queueing and inventory models of Chaps. 3 and 4 required interarrival times, service times, demand sizes, etc.

Properties of Ran. Num. (1/3)

Introduction (2/3)

- A random number generator (RNG) is any mechanism that produces independent random numbers. The term independent implies that the probability of producing any given random number remains the same each time a number is produced.

Properties of Ran. Num. (1/3)

Introduction (3/3)

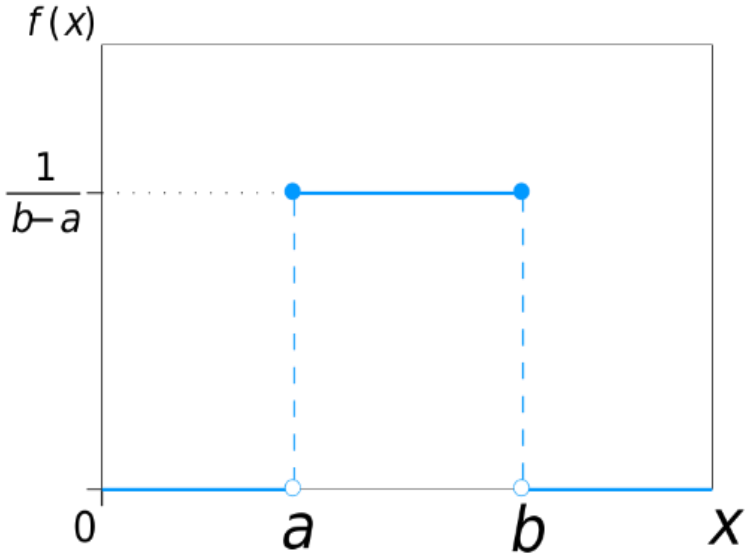
- A sequence of random numbers $R_1, R_2, \ldots,$ must have two important statistical properties:
  - Uniformity,
  - Independence.

Properties of Ran. Num. (2/3)

Uniformity (1/3)

- Random Number, R_i, must be independently drawn from a uniform distribution with probability density function (pdf):

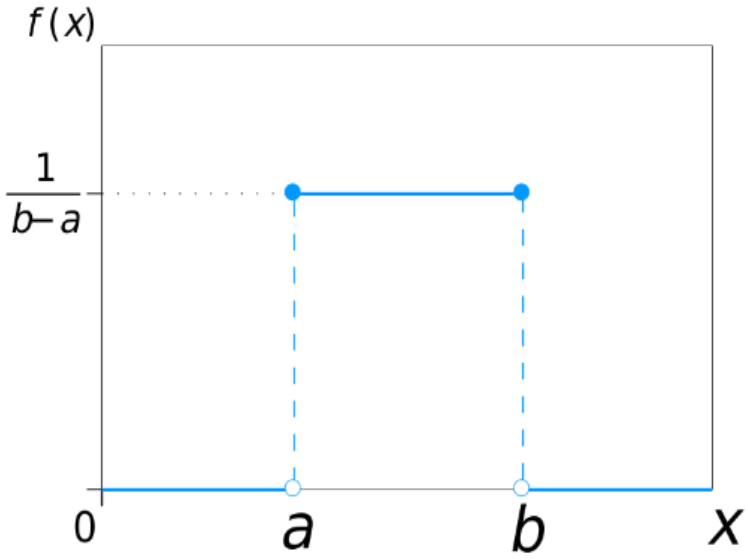
$$f(x) = \begin{cases} \dfrac{1}{b-a} & for\ x \in [a,b] \\ 0 & otherwise \end{cases}$$



Properties of Ran. Num. (2/3)

Uniformity (2/3)

$$f(x) = \begin{cases} \dfrac{1}{b-a} & for\ x \in [a,b] \\ 0 & otherwise \end{cases}$$



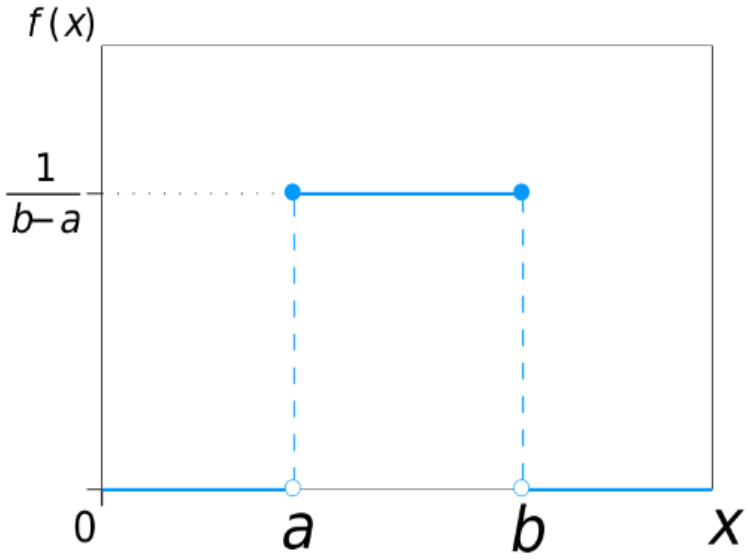
$$\text{Mean} = E(X) = \frac{a+b}{2}$$

$$\text{Varaince} = V(X) = \frac{(b-a)^2}{12}$$

Properties of Ran. Num. (2/3)

Uniformity (2/3)

$$f(x) = \begin{cases} \dfrac{1}{b-a} & for\ x \in [a,b] \\ 0 & otherwise \end{cases}$$



$$\text{Mean} = E(X) = \frac{a+b}{2}$$

$$\text{Varaince} = V(X) = \frac{(b-a)^2}{12}$$

If we select $a = 0$ and $b = 1$

Standard Uniform Distribution

Properties of Ran. Num. (2/3)

Uniformity (3/3)

- If we select $a = 0$ and $b = 1$

$$f(x) = \begin{cases} 1 & \text{for } x \in [a, b] \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Mean} = E(X) = \frac{1}{2}$$

$$\text{Varaince} = V(X) = \frac{1}{12}$$

Properties of Ran. Num. (3/3)

Independence (1/2)

- In true random numbers, if you have a random number, the next random number is always unpredictable.
- Random processes are said to be nondeterministic.
- While, computers, on the other hand, are deterministic. Therefore, arithmetic methods are used to generate random numbers to be implemented on the computer.

Properties of Ran. Num. (3/3)

Independence (2/2)

- By definition, a true random number cannot be predicted. Numbers produced by a random number generator are calculated, and a calculated number is predictable. Thus, the numbers created by a random number generator are often referred to as pseudorandom numbers (PRNs).

Generation of PRNs (1/3)

Introduction

There are numerous methods that can be used to generate random values. Some important considerations concerning these methods or routines include **speed** of execution, **portability**, **replicability**, numbers produced must be statistically **independent**, **period** of the produced random sequence should be **long**, and technique used should **not require large memory** space.

Generation of PRNs (2/3)

The Routine Should Be Fast

Individual computations are inexpensive, but a simulation may require many millions of random numbers.

Portable To Different Computers

Ideally to different programming languages. This ensures the program produces same results.

Generation of PRNs (3/3)

Have Sufficiently Long Cycle

The cycle length, or period represents the length of random number sequence before previous numbers begin to repeat in an earlier order.

Replicable

Given the starting point, it should be possible to generate the same set of random numbers, completely independent of the system that is being simulated.

Techniques for PRNG (1/3)

- The Middle-Square (*midsquare*) Method.
- Linear-Congruential Generation (LCG).
- Mersenne Twister (MT).
- 64-bit MELG.

The following link list many algorithms for pseudorandom number generators.

https://en.wikipedia.org/wiki/List_of_random_number_generators

Techniques for PRNG (2/3)

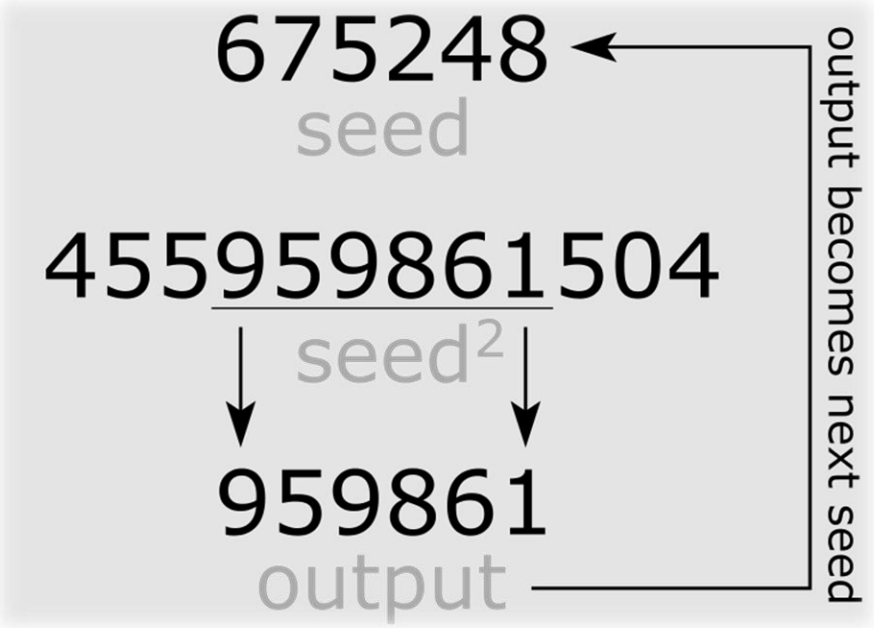
Middle-Square (*midsquare*) Method (1/7)

- The first such arithmetic generator, proposed in the 1949s, is the famous ***midsquare*** method.
- To generate a sequence of *n*-digit pseudorandom numbers, an *n*-digit starting value is created and squared, producing a *2n*-digit number. If the result has fewer than *2n* digits, leading zeroes are added to compensate. The middle *n* digits of the result would be the next number in the sequence and returned as the result. This process is then repeated to generate more numbers.

Techniques for PRNG (2/3)

Middle-Square (*midsquare*) Method (2/7)

- The **seed** value used to initialize an PRNG.
- Here, we used 6-digits PRN.



Techniques for PRNG (2/3)

Middle-Square (*midsquare*) Method (3/7)

- The value of *n* must be even in order for the method to work. If the value of *n* is odd then there will not necessarily be a uniquely defined 'middle *n*-digits' to select from.

Techniques for PRNG (2/3)

Middle-Square (*midsquare*) Method (4/7)

- Consider the following: If a 3-digit number is squared it can yield a 6-digit number (eg: $540^2 = 291600$).
- If there were to be a middle three digit that would leave $6 - 3 = 3$ digits to be distributed to the left and right of the middle. It is impossible to evenly distribute these digits equally on both sides of the middle number and therefore there are no 'middle digits.' It is acceptable to pad the seeds with zeros to the left in order to create an even valued *n*-digit (eg: 540 → 0540).

Techniques for PRNG (2/3)

Middle-Square (*midsquare*) Method (5/7)

- For a generator of *n*-digit numbers, the period can be no longer than 8^n.
- If the middle *n* digits are all zeroes, the generator then outputs zeroes forever.
- If the first half of a number in the sequence is zeroes, the subsequent numbers will be decreasing to zero. While these runs of zero are easy to detect, they occur too frequently for this method to be of practical use.

Techniques for PRNG (2/3)

Middle-Square (*midsquare*) Method (6/7)

- The middle-squared method can also get stuck on a number other than zero. For $n = 4$, this occurs with the values 0100, 2500, 3792, and 7600. Other seed values form very short repeating cycles, e.g., 0540 → 2916 → 5030 → 3009.
- These phenomena are even more obvious when $n = 2$, as none of the 100 possible seeds generates more than 14 iterations without reverting to 0, 10, 50, 60, or a 24↔57 loop.

Techniques for PRNG (2/3)

Middle-Square (*midsquare*) Method (7/7)

- Example with seed = 7182

i	Z_i	U_i	Z_i^2
0	7182	—	51,581,124
1	5811	0.5811	33,767,721
2	7677	0.7677	58,936,329
3	9363	0.9363	87,665,769
4	6657	0.6657	44,315,649
5	3156	0.3156	09,960,336
.	.	.	.
.	.	.	.
.	.	.	.

Video Lectures

All Lectures: https://www.youtube.com/playlist?list=PLxIvc-MGOs6geFJmdvDOIN5zE89-Hq8Ij

Lecture #7: https://www.youtube.com/watch?v=veIPkhICi6w&list=PLxIvc-MGOs6geFJmdvDOIN5zE89-Hq8Ij&index=21

https://www.youtube.com/watch?v=ERLT44VFN8w&list=PLxIvc-MGOs6geFJmdvDOIN5zE89-Hq8Ij&index=22

Thank You

Dr. Ahmed Hagag

ahagag@fci.bu.edu.eg