CryptoRand: Interactive Web Application for Exploring Cryptographic Pseudo-Random Number Generators

Author: Anugrah Singh (1AY23CS031) **Course:** Discrete Mathematics Structures **Instructor:** Prof. Gayathri
Date: June 8, 2025

1. Abstract

CryptoRand is an interactive web application designed to demonstrate and educate users on the application of discrete mathematics principles in cryptographic pseudo-random number generators (PRNGs). The project allows users to generate sequences from PRNGs like the Linear Congruential Generator (LCG) and Blum Blum Shub (BBS), visualize their outputs, and learn underlying mathematical concepts through an AI-powered tutor (Gemma 3). It aims to be an educational, visually appealing, and cutting-edge showcase of how discrete mathematics is fundamental to cryptography.

2. Introduction

2.1 Motivation

The motivation behind CryptoRand is to bridge the gap between theoretical discrete mathematics concepts and their practical applications in cryptography, specifically in the domain of pseudo-random number generation. Many students find it challenging to grasp these concepts without interactive and visual tools. This project serves as a hands-on platform for learning and experimentation.

2.2 Problem Statement

Understanding the intricacies of PRNGs, their strengths, weaknesses, and the discrete mathematical principles they are built upon (e.g., modular arithmetic, number theory) can be complex. CryptoRand addresses this by providing an accessible environment to explore these topics interactively.

2.3 Project Goals and Objectives

- To implement common PRNGs (LCG, BBS) based on discrete mathematics.
- To allow users to interactively adjust PRNG parameters and observe outcomes.
- To provide clear visualizations (histograms, scatter plots) of PRNG outputs.
- To integrate an AI tutor (Gemma 3) for explaining discrete math concepts and their cryptographic relevance.
- To create engaging educational modules and exercises.
- To develop a user-friendly, visually appealing web application suitable for public demonstration and educational purposes.

3. Project Description

CryptoRand is a web application that offers a comprehensive platform for exploring PRNGs. Users can:

- **Generate PRNG Sequences:** Select a PRNG (LCG or BBS), input parameters like seed, modulus, multiplier, and generate sequences of pseudo-random numbers.

- **Visualize Outputs:** View real-time visualizations of the generated sequences, such as histograms to see the distribution of numbers and scatter plots to identify patterns or cycles.
- **Learn with an AI Tutor:** Interact with an AI tutor powered by a locally hosted Gemma 3 LLM. The tutor can explain discrete mathematics concepts (e.g., modular arithmetic, finite fields, quadratic residues, entropy), discuss their application in the chosen PRNG, analyze PRNG outputs, and answer user queries.
- **Explore Cryptographic Demonstrations:** See simple examples of how PRNGs are used in cryptographic contexts, such as key generation.
- **Engage with Educational Modules:** Work through guided lessons and interactive exercises that reinforce understanding of PRNGs and related mathematical principles.

4. Technical Details

4.1 Tech Stack

- **Frontend:** Vite with React (for a fast, dynamic UI)
- **Styling:** Tailwind CSS (for a modern, responsive design)
- **Visualizations:** Chart.js
- **Backend:** Flask (Python) for PRNG computations, API handling, and AI integration.
- **AI Integration:** Gemma 3, locally hosted via LM Studio, accessed through an API.
- **Mathematical Computations:** NumPy/SciPy (primarily in the backend).
- **Version Control:** Git

4.2 PRNG Implementations

4.2.1 Linear Congruential Generator (LCG)

- **Mathematical Formula:** `X_n+1 = (a * X_n + c) mod m`
  - `X_n` is the current state (or number).
  - `X_n+1` is the next state.
  - `a` is the multiplier.
  - `c` is the increment.
  - `m` is the modulus.
  - `X_0` is the seed.
- **Discrete Math Concepts:** Modular arithmetic is the core concept. The choice of `a`, `c`, `m`, and the seed significantly impacts the period and quality of the generated sequence.
- **Implementation:** The Flask backend implements the LCG algorithm, allowing users to set `a`, `c`, `m`, and the seed via the frontend.

4.2.2 Blum Blum Shub (BBS)

- **Mathematical Formula:** `X_n+1 = X_n^2 mod M`
  - `M = p * q`, where `p` and `q` are large prime numbers, both congruent to 3 (mod 4).
  - The seed `X_0` must be coprime to `M`.
  - The output is typically the least significant bit of `X_n+1`.
- **Discrete Math Concepts:** Quadratic residues, modular arithmetic, prime numbers, number theory (specifically properties of primes congruent to 3 mod 4), and finite fields. The security of BBS relies on the difficulty of factoring `M`.

- **Implementation:** The Flask backend implements BBS, allowing users to (potentially) set the seed, or use pre-defined `p` and `q` for `M`. The generation of suitable primes `p` and `q` is a critical part.

4.3 Visualizations

Chart.js is used to create:

- **Histograms:** To show the frequency distribution of the generated numbers. A good PRNG should produce a relatively uniform distribution.
- **Scatter Plots:** (e.g., plotting `X_n` vs `X_n+1`) To help visualize patterns, correlations, or the length of cycles in the PRNG output. These can highlight weaknesses in certain PRNG configurations.

4.4 AI Integration (Gemma 3)

- **Role:** The AI tutor, powered by Gemma 3 (hosted locally via LM Studio), provides:
  - Explanations of discrete math concepts relevant to PRNGs (e.g., modular arithmetic, prime numbers, quadratic residues, entropy).
  - Insights into how PRNGs are used in cryptography (e.g., for generating keys, nonces).
  - Real-time analysis and interpretation of PRNG outputs and their statistical properties.
- **Integration:** The Flask backend communicates with the LM Studio API to send user queries or PRNG data to Gemma 3 and relays the AI's responses back to the frontend.

4.5 Statistical Tests

- The project includes statistical tests, such as the chi-square test, to analyze the randomness of the generated sequences. The results of these tests are presented to the user, often with AI-powered explanations of their significance.

4.6 Source Code Repository

- The complete source code for the CryptoRand project is available on GitHub: https://github.com/Anugrah-Singh/dms_project

5. Discrete Mathematics Concepts Applied

CryptoRand heavily relies on and demonstrates several core concepts from discrete mathematics:

- **Modular Arithmetic:** Fundamental to both LCG (definition) and BBS (definition and output generation). The `mod` operator is central.
- **Number Theory:**
  - **Prime Numbers:** Crucial for BBS (selection of `p` and `q`).
  - **Greatest Common Divisor (GCD):** Important for selecting seeds and parameters in LCG and BBS (e.g., seed coprime to `M` in BBS).
  - **Congruences:** Used in defining parameters for BBS (primes `p, q ≡ 3 (mod 4)`).
- **Quadratic Residues:** The core mathematical structure behind the BBS generator.
- **Finite Fields:** The operations in BBS occur within a finite field `Z_M`.
- **Sequences and Recurrence Relations:** PRNGs are defined by recurrence relations (`X_n+1` depends on `X_n`).
- **Set Theory:** Understanding the range of numbers that can be generated (`{0, 1, ..., m-1}` for LCG).

- **Combinatorics and Probability:** Implicit in analyzing the "randomness" and distribution of PRNG outputs. Statistical tests are based on probabilistic expectations.
- **Graph Theory:** (Potentially) Visualizing state transitions in PRNGs can be represented as graphs, helping to understand cycle lengths.

6. Educational Value

CryptoRand aims to be a powerful educational tool by:

- **Interactive Learning:** Allowing users to experiment with parameters and see immediate visual and statistical results, fostering an intuitive understanding.
- **Concept Reinforcement:** The AI tutor provides on-demand explanations, connecting the practical PRNG behavior back to theoretical discrete math concepts.
- **Guided Modules:** Structured lessons and exercises help users systematically explore different aspects of PRNGs and cryptography.
- **Accessibility:** Making complex topics more approachable for students and enthusiasts who may not be experts in cryptography or advanced mathematics.

7. User Interface and Experience

- **Design Philosophy:** The UI is designed to be clean, intuitive, and visually engaging, encouraging exploration and learning.
- **Technology:** Tailwind CSS is used for styling, ensuring a modern look and feel, and responsiveness across different devices.
- **User-Friendliness:** The application is designed for ease of use, even for those new to PRNGs, making it suitable for public demonstrations.

8. Challenges and Future Work (Optional Section)

8.1 Challenges

- Integrating the local LLM (Gemma 3 via LM Studio) smoothly with the Flask backend and ensuring real-time, relevant responses.
- Designing effective visualizations that clearly convey the properties (and flaws) of PRNGs.
- Creating educational content that is both accurate and easy to understand for a diverse audience.

8.2 Future Work

- Implement additional PRNGs (e.g., Mersenne Twister, cryptographic stream ciphers).
- Expand the range of statistical tests for randomness.
- Develop more advanced cryptographic demonstrations (e.g., simple encryption/decryption using PRNG-generated keys).
- Allow users to save and compare results from different PRNG configurations.
- Explore cloud deployment options for wider accessibility.

9. Conclusion

CryptoRand successfully demonstrates the critical role of discrete mathematics in the design and analysis of pseudo-random number generators. By combining interactive PRNG implementations, dynamic visualizations,

and AI-driven explanations, the project provides an effective and engaging platform for learning about these foundational cryptographic tools. It serves as a practical application of concepts such as modular arithmetic, number theory, and quadratic residues, making them more tangible and understandable. The project meets its goal of being an educational, visually appealing, and cutting-edge tool for students and anyone interested in the intersection of mathematics and cryptography.

10. References (If applicable)

- [List any books, papers, or significant online resources consulted]

11. Appendix (Optional)

- [Could include key code snippets, detailed architectural diagrams, or extended mathematical derivations if required for the assignment]

This report structure should provide a comprehensive overview of your project for your assignment. Remember to fill in the placeholders like "[Your Name/Student ID]" and "[Your Instructor's Name]". You might also want to add specific examples or screenshots if your assignment guidelines allow for it.