Lagont's Ant Encryption Algorithm

Introduction

Description:

In this project I propose an encryption algorithm based on a cellular automaton. A successful encryption algorithm is one that effectively protects data from unauthorized access while allowing authorized parties to access it. It should be difficult to crack, even with advanced computing resources, and provide confidentiality, integrity, and authenticity to the data. Additionally, it should be resistant to attacks such as brute-force attacks, known plaintext attacks, and chosen plaintext attacks.

Why system was chosen:

To create this encryption algorithm, I used Langton's ant cellular automata due to its pseudo random nature.

Model concept:

Langton's ants is a two-dimensional simulation where an ant is placed on a matrix and allowed to move according to specific rules. Given a set of rules, the ants develop a pattern and three types of emergent behavior. The movement of the ant is determined by simple rules based on the color of the cell it is on. These rules dictate the direction the ant moves in and the color of the cell it leaves behind.

Three emergent behaviors are expected –

1. Simplicity – where it creates simplistic and often symmetric patterns early in its run time.
2. Chaos – where irregular patterns are created in the next 10,000 steps.
3. Emergent – where a "highway" pattern is created, and the ant moves in the same direction.

In this example, an encryption key is generated given the initial input conditions provided by the user (starting state of the matrix, ant positions, ant rules), the emergent behavior is then used as a key, the message is XORed with the key to be encrypted, and then XORed again to be decrypted. The most pseudo random behavior would be simplicity and chaos emergent patterns, so step number should be less than 10,000.

Motivation

Problem(s) to be solved:

The Langton's ant encryption algorithm is designed to provide a secure method of encryption that is difficult to crack. The Langton's ant encryption algorithm is designed to be resistant to these types of attacks by using a key that is based on the unpredictable and pseudo random behavior of Langton's ant.

Why simulation is necessary:

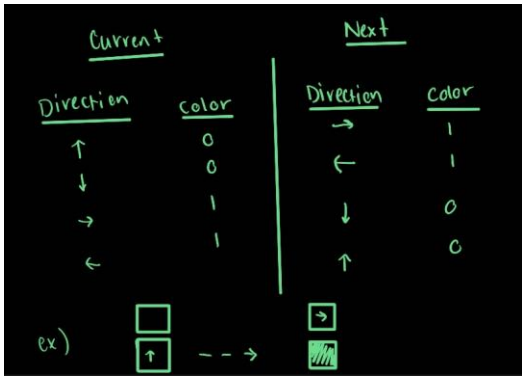
Simulation is necessary to test the effectiveness of the Langton's ant encryption algorithm. By simulating the movement of Langton's Ant on a grid, we can test the algorithm under various conditions to determine its effectiveness as an encryption method. We need to simulate different emergent behaviors because the best key would be the ones without patterns, and as stated in the introduction, emergent highway pattern or symmetric patterns can appear depending on the step size.

Background

The basic rules of the simulation is that a directed ant is placed randomly on the matrix, then:

1. When it arrives at a white square, it turns right and flips the color of the square to black.
2. When it arrives at a black square, it turns left and flips the color of the square to white.

Diagram:



Assumptions:

In this implementation, we assume one ant, one set of rules, and 2 colors.

In terms of the success of the encryption algorithm, we assume that the movement of Langton's ant on a grid is unpredictable and cannot be easily replicated.

Course Concepts:

This simulation uses course concepts of cellular automata, rule based modeling, and synchronous updates.

Verification and Scenarios

Beside visual inspection, there are a few ways to determine the correctness of Langton's Ant simulation.

- There are several known patterns that can emerge from the simulation. For example, after a certain number of steps, the ant may form a highway pattern or a spiral pattern. By running the simulation for a sufficient number of steps and checking for the presence of these known patterns, I can verify that the simulation is behaving correctly.
- By analyzing the distribution of cell states (black or white) at different points in the simulation. If the simulation is working correctly, the distribution should be roughly equal between black and white cells.
- By comparing the results of my simulation to the results of previous implementations, I can validate that my simulation is consistent with the expected behavior of Langton's Ant.
- Testing different initial configurations and comparing the resulting patterns is another way to validate the correctness of the simulation. Langton's Ant is known to be highly sensitive to its initial conditions. By testing different initial configurations and comparing the resulting patterns, I can validate that the simulation is correctly capturing the sensitivity to initial conditions.

To determine the validity of the encryption algorithm, I can use the following steps:

1. Generate a key using the initial conditions of the Langton's Ant simulation.
2. Encrypt an image using the generated key.
3. Generate the same key with the same initial conditions.
4. Decrypt the encrypted image using the generated key.
5. Verify that the decrypted image is the same as the original input image.

If the decrypted image is the same as the original input image, then I can be confident that the encryption algorithm is working correctly.

I will also change the input used to generate the key slightly to show that the decrypted message is not the same as the original.

As seen in the code, the verification is successful given real user input of a message and key.

Results

The results are shown and commented on in the Jupyter Notebook.

Evaluation

The results of the simulation showed that it was able to encrypt messages effectively, and even small changes in the key parameters would result in significant changes in the encrypted message. However, the algorithm has some limitations, such as its vulnerability to brute force attacks if the key is known, and its reliance on a chaotic system that is difficult to predict.

Challenges: one challenge with Langton's Ant Encryption is the need for a large key matrix to encrypt relatively small messages. This can lead to slower encryption and decryption times as well as increased memory usage. As the size of the message increases, the size of the key matrix must also increase proportionally, which can be a limitation in practical applications. One possible approach to address this challenge is to use a technique called key stretching, which involves applying a one-way function to the key to create a larger, pseudorandom key.

Alternate approaches: There are many other encryption algorithms that can be used instead of Langton's ant encryption. For example, symmetric key encryption algorithms like AES and Blowfish are commonly used for secure communication. Public key encryption algorithms like RSA are also widely used for secure communication over the internet. Furthermore, different cellular automata can be used to generate keys, like Game of Life.

Conclusion

This project utilized Langont's ant simulation for an encryption algorithm successfully. Users can input a message and key parameters, then a key is generated to encrypt and decrypt the message. As part of this project, I also modeled the different emergent behaviors of Langont's ant to find a good step size for key parameters. In the future, I hope to explore the possibilities of collisions with this algorithm. I also hope to add more key parameters, including different rules, different colors, and more ants.

Citation

The idea of using Langton's ant as an encryption algorithm is mentioned in this paper, but the implementation differs:

https://iopscience.iop.org/article/10.1088/1742-6596/1840/1/012019/pdf

Information about Langton's ant and its variations can be found here:

https://en.wikipedia.org/wiki/Langton%27s_ant