Застосування алгоритму index-calculus для дискретного логарифмування

1 Мета

Ознайомлення з алгоритмом дискретного логарифмування index-calculus. Програмна реалізація цього алгоритму та визначення його переваг, недоліків та особливостей застосування. Практична оцінка складності роботи та порівняння різних реалізацій цього алгоритму.

2 Постановка задачі

Написати програму, яка імплементує алгоритм index-calculus для пошуку дискретного логарифму груп типу Z_p

3 Хід роботи

3.1 План

1. Ознайомлення з алгоритмом index-calculus
2. Імплементація алгоритму
   (а) Формування факторної бази
   (б) Генерація рівнянь для СЛР
   (в) Вирішення СЛР
   (г) Обчислення самого дискретного логарифму на основі рішення СЛР
3. Підняття Docker контейнеру

3.2 Проблеми які виникнули під час імплементації алгоритму

Одна з основних проблем це вирішення самої СЛР. Спочатку були спроби піти тривіальним шляхом та за допомогою **numpy** вирішити СЛР пошуком оберненої матриці, тобто: $X = A^{-1} \cdot B$. Однак, це не спрацювало оскільки матриця A не завжди була квадратною, а отже не завжди можна було знайти обернену матрицю. Були спроби підвести СЛР так, щоб матриця A завжди була квадратною, однак зробити це в загальному випадку не вийшло. Тому далі було прийняте рішення використовувати модифікаю алгоритму Гаусса. Тут теж не все так просто, оскільки запрограмувати це як вияснилось зовсім не легко.
Розпаралелити генерацію рівнянь в СЛР також не вийшло, оскільки відповідь завжди була не правильною і знайти в чому проблема, на жаль, не вдалось.

3.3 Кілька слів про саму програму

По не зрозумілим причинам, програма працює коректно для чисел довжина яких не перевищує **10**. Також, відповіді для задач типу 2 знаходяться неправильно, хоча для задач типу 1 майже завжди коректно, тому порівняти час обчислення дискретного логарифму для задач різних типів не вийшло.

4 Приклад роботи програми

Testing Index-Calculus with a = 179, b = 97, p = 191 digit length: 3, IC result: x = 168 (took 0.001 seconds)

Testing Index-Calculus with a = 3086, b = 2576, p = 3617 digit length: 4, IC result: x = 1038 (took 0.002 seconds)

Testing Index-Calculus with a = 606, b = 19755, p = 33773 digit length: 5, IC result: x = 9717 (took 0.004 seconds)

Testing Index-Calculus with a = 69366, b = 534740, p = 889081 digit length: 6, IC result: x = 630451 (took 0.014 seconds)

Testing Index-Calculus with a = 3842476, b = 6675652, p = 8043979 digit length: 7, IC result: x = 7268042 (took 0.04 seconds)

Testing Index-Calculus with a = 66830006, b = 51535128, p = 87321277 digit length: 8, IC result: x = 26090344 (took 0.12 seconds)

5 Замір часу роботи

На графіку показано залежність часу обчислень від порядку числа.

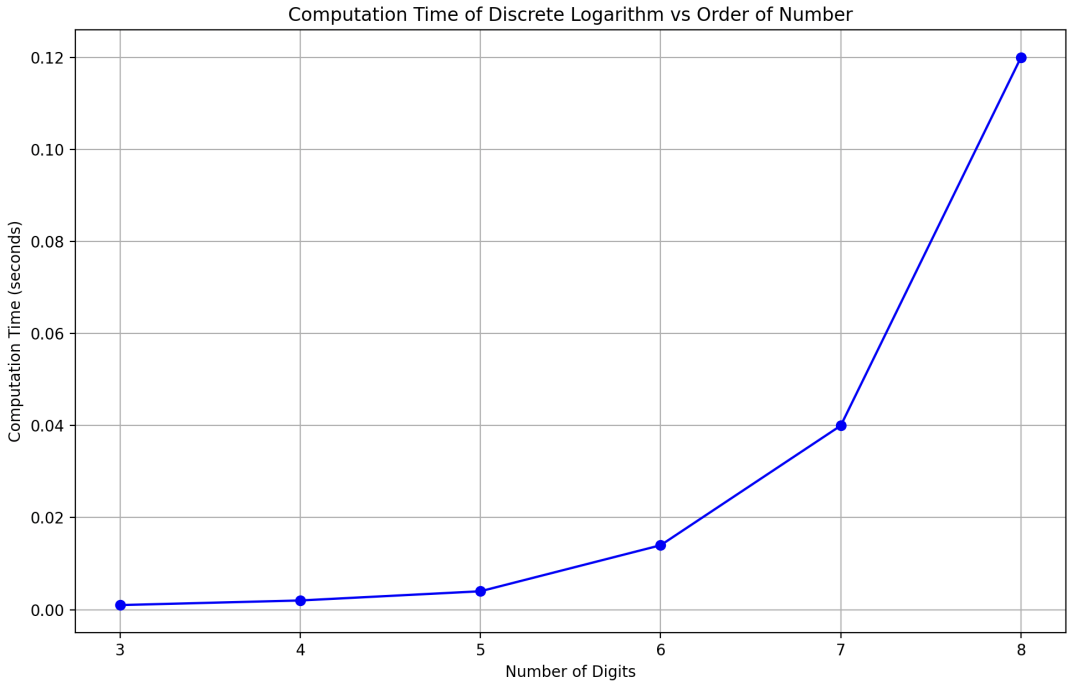

Рис. 1: час роботи

6 Запуск Docker контейнера

```
docker run aveeept/nta-lab-3:0.2 python3 discrete_logarithm.py a b p
```

7 Висновок

У цій роботі було розроблено програму для розв'язку задачі дискретного логарифму використовуючи index-calculus, автоматизовано заміри часу роботи розробленого алгоритму. На жаль,

імплементувати алгоритм щоб він коректно працював для всіх типів задач, а також для чисел довжина яких перевищує 10 не вийшло, однак було проведено чимала кількість спроб для пошуку проблеми та її виправлення. З плюсів, вийшло розібрати як запрограмувати модифікацію алгоритму Гаусса для розв'язку СЛР.