КОМП'ЮТЕРНИЙ ПРАКТИКУМ 1

з курсу

ТЕОРЕТИКО-ЧИСЛОВІ АЛГОРИТМИ В КРИПТОЛОГІЇ

Пошук канонічного розкладу великого числа, використовуючи відомі методи факторизації

Мета роботи

Практичне ознайомлення з різними методами факторизації чисел, реалізація цих методів і їх порівняння. Виділення переваг, недоліків та особливостей застосування алгоритмів факторизації. Застосування комбінації алгоритмів факторизації для пошуку канонічного розкладу заданого числа.

Хід роботи

Напишемо програми, що реалізовують такі алгоритми:

(а) тест на простоту Міллера-Рабіна

(б) метод пробних ділень

(в) ρ-метод Полларда

(г) метод Брілхарта-Моррісона

З написанням алгоритмів а,б і в проблем не було, а от з метом Брілхарта-Моррісона виникли деякі труднощі: написання коду для розв'язання СЛР і його оптимізація. Ретельно проаналізувавши додаткові матеріали, вдалося прийти до більш-менш оптимального варіанту.

Застосуємо алгоритм написаний на кроці 3 до чисел 691534156424661573 (4 варіант)

```
Program started,factored number = 691534156424661573
Trivial division found factors [3], time passed from start: 0.0
Pollard rho algorithm found factors 7877, time passed from start: 0.0
Brilhart morrison algorithm found factors [33303617], time passed from start: 56.93827295303345
Variant factors: [3, 7877, 33303617, 878699]

Process finished with exit code 0
```

і 341012868237902669 (9 варіант)

```
Program started,factored number = 341012868237902669
Trivial division found factors [23], time passed from start: 0.0
Pollard rho algorithm found factors 1549, time passed from start: 0.0010006427764892578
Brilhart morrison algorithm found factors [937033], time passed from start: 31.791518688201904
Variant factors: [23, 1549, 937033, 10214959]

Process finished with exit code 0
```

Перевіримо

Целое число
691534156424661573

Факторизация
$3 \cdot 7877 \cdot 878699 \cdot 33303617$

Целое число
341012868237902669

Факторизация
$23 \cdot 1549 \cdot 937033 \cdot 10214959$

Тепер застосуємо алгоритми ρ-метод Полларда та метод Брілхарта-Моррісона до наступних чисел і виміряємо час

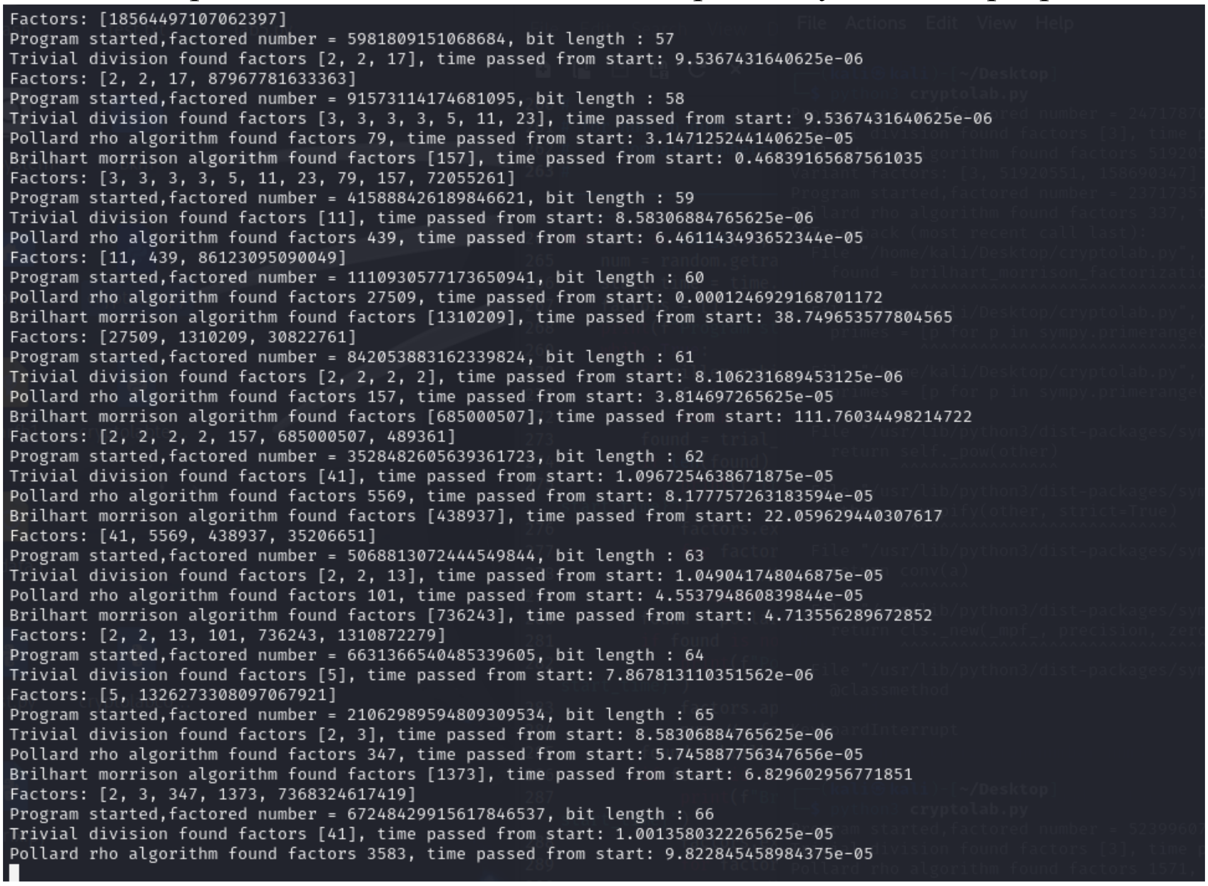
```
Factoring 3009182572376191
Brilhart morisson found factor [30091489] in 4.187784194946289 seconds
Pollard rho found factor 100001119 in 0.004878282546997070 seconds
Factoring 1021514194991569
Brilhart morisson found factor [100001791] in 209.7772765159607 seconds
Pollard rho found factor 10214959 in 0.002259492874145508 seconds
Factoring 4000852962116741
Brilhart morisson found factor [40007321] in 450.7394814491272 seconds
Pollard rho found factor 100003021 in 0.0037262439727783203 seconds
Factoring 15196946347083
Brilhart morisson found factor [3] in 1.5785093307495117 seconds
Pollard rho found factor 3 in 3.7670135498046875e-05 seconds
Factoring 499664789704823
Brilhart morisson found factor [33303617] in 136.13421034812927 seconds
Pollard rho found factor 33303617 in 0.001489400863647461 seconds
Factoring 269322119833303
Brilhart morisson found factor [24779017] in 93.00886917114258 seconds
Pollard rho found factor 10868959 in 0.0030236244201660156 seconds
Factoring 679321846483919
Brilhart morisson found factor [28065119] in 134.93713426589966 seconds
Pollard rho found factor 28065119 in 0.0036869049072265625 seconds
Factoring 96267366284849
Brilhart morisson found factor None in 164.34762954711914 seconds
Pollard rho found factor 962623 in 0.0008475780487060547 seconds
Factoring 61333127792637
Brilhart morisson found factor [3] in 2.0112850666046143 seconds
Pollard rho found factor 3 in 4.172325134277344e-05 seconds
Factoring 2485021628404193
Brilhart morisson found factor None in 1057.0169885158539 seconds
Pollard rho found factor 100002967 in 0.004319667816162109 seconds

Process finished with exit code 0
```

Бачимо, що ρ-метод Полларда знаходить дільники швидше, оскільки він не залежить від сторонніх чинників, такі як вирішення СЛР і побудова факторної бази. Також це алгоритм типу Las-Vegas, тому він не гарантовано поверне дільника, але якщо знайде, то він точно правильний.

Варто зауважити, що алгоритм Брілхарта-Моррісона не знайшов дільник для останього числа через замаленьку факторну базу. Вона шукалась при значеннях а = $\frac{1}{\sqrt{2}}, 1, \frac{\sqrt{3}}{\sqrt{2}}, \sqrt{2}$

Оцінимо порядок числа, яке ще піддається розкладу нашою програмою



```
Factors: [18564497107062397]
Program started,factored number = 5981809151068684, bit length : 57
Trivial division found factors [2, 2, 17], time passed from start: 9.5367431640625e-06
Factors: [2, 2, 17, 87967781633363]
Program started,factored number = 91573114174681095, bit length : 58
Trivial division found factors [3, 3, 3, 3, 5, 11, 23], time passed from start: 9.5367431640625e-06
Pollard rho algorithm found factors 79, time passed from start: 3.147125244140625e-05
Brilhart morrison algorithm found factors [157], time passed from start: 0.46839165687561035
Factors: [3, 3, 3, 3, 5, 11, 23, 79, 157, 72055261]
Program started,factored number = 415888426189846621, bit length : 59
Trivial division found factors [11], time passed from start: 8.58306884765625e-06
Pollard rho algorithm found factors 439, time passed from start: 6.461143493652344e-05
Factors: [11, 439, 86123095090049]
Program started,factored number = 1110930577173650941, bit length : 60
Pollard rho algorithm found factors 27509, time passed from start: 0.0001246929168701172
Brilhart morrison algorithm found factors [1310209], time passed from start: 38.749653577804565
Factors: [27509, 1310209, 30822761]
Program started,factored number = 842053883162339824, bit length : 61
Trivial division found factors [2, 2, 2, 2], time passed from start: 8.106231689453125e-06
Pollard rho algorithm found factors 157, time passed from start: 3.814697265625e-05
Brilhart morrison algorithm found factors [685000507], time passed from start: 111.76034498214722
Factors: [2, 2, 2, 2, 157, 685000507, 489361]
Program started,factored number = 3528482605639361723, bit length : 62
Trivial division found factors [41], time passed from start: 1.0967254638671875e-05
Pollard rho algorithm found factors 5569, time passed from start: 8.177757263183594e-05
Brilhart morrison algorithm found factors [438937], time passed from start: 22.059629440307617
Factors: [41, 5569, 438937, 35206651]
Program started,factored number = 5068813072444549844, bit length : 63
Trivial division found factors [2, 2, 13], time passed from start: 1.049041748046875e-05
Pollard rho algorithm found factors 101, time passed from start: 4.553794860839844e-05
Brilhart morrison algorithm found factors [736243], time passed from start: 4.713556289672852
Factors: [2, 2, 13, 101, 736243, 1310872279]
Program started,factored number = 6631366540485339605, bit length : 64
Trivial division found factors [5], time passed from start: 7.867813110351562e-06
Factors: [5, 1326273308097067921]
Program started,factored number = 21062989594809309534, bit length : 65
Trivial division found factors [2, 3], time passed from start: 8.58306884765625e-06
Pollard rho algorithm found factors 347, time passed from start: 5.745887756347656e-05
Brilhart morrison algorithm found factors [1373], time passed from start: 6.829602956771851
Factors: [2, 3, 347, 1373, 7368324617419]
Program started,factored number = 67248429915617846537, bit length : 66
Trivial division found factors [41], time passed from start: 1.0013580322265625e-05
Pollard rho algorithm found factors 3583, time passed from start: 9.822845458984375e-05
```

Після декількох тестів, дійшли до висновку, що наш алгоритм, спроможний факторизувати 66 бітне число (з похибкою декілька бітів)

Висновок

Проведений аналіз різних методів факторизації чисел дозволив нам розглянути їхні переваги, недоліки та особливості застосування. Метод Міллера-Рабіна, який використовується для тестування на простоту, виявився ефективним і швидким. Він дозволяє виявити простоту числа з високою ймовірністю, зменшуючи ризик помилкового визначення простоти. Метод пробних ділень також є досить ефективним, особливо для невеликих чисел. Однак він може стати непрактичним для великих чисел через велику обчислювальну складність. Р-метод Полларда виявився надзвичайно швидким та ефективним для великих чисел, оскільки він не потребує побудови факторної бази і залежить від особливостей числа. Однак метод Брілхарта-Моррісона, хоча потенційно ефективний, може вимагати значної оптимізації та уточнення параметрів для досягнення задовільних результатів. В деяких випадках він може навіть не знайти дільника через недостатню факторну базу. Крім того, застосування комбінації алгоритмів факторизації може бути ефективним підходом для складних чисел, дозволяючи використовувати переваги кожного методу. У результаті проведених експериментів було встановлено, що розроблений алгоритм факторизації може успішно розкладати 66-бітні числа з прийнятною похибкою, що свідчить про його високу ефективність.