ФАКТОРИЗАЦИЯ ЧИСЕЛ РО-МЕТОДОМ ПОЛЛАРДА

Кузнецова Арина 6373

Ро-алгоритм Полларда

В 1975 году Поллард опубликовал статью, в которой он, основываясь на алгоритме обнаружения циклов Флойда, изложил идею алгоритма факторизации чисел, работающего за время, пропорциональное $N^{1/4}$.

С его помощью было разложено на множители число Ферма $F_8 = 2^{256} + 1$.

Оригинальная версия алгоритма

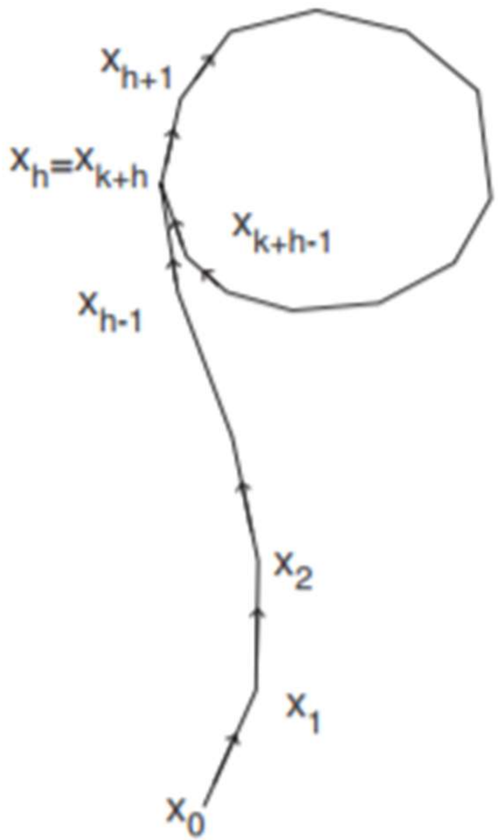
Возьмем некоторое случайное отображение

$f\colon Z_n \to Z_n$, которое сгенерирует некоторую случайную последовательность $x_0, x_1, x_2...$ где $x_i = f(x_{i-1})$. Обычно берётся многочлен $f(x) = x^2$+1.

Функция f имеет не более, чем n значений, поэтому последовательность зациклится.

$$x_{k+h} = x_h$$

h называется индексом вхождения, k - длиной цикла.



Оригинальная версия алгоритма

Рассмотрим теперь алгоритм подробнее.

При выбранном многочлене $f(x) = x^2 + 1 \ (mod\ n)$ рассматриваем $x_i = f(x_{i-1})$. Причём i - индекс элемента перед зацикливанием последовательности.

И для всех $j < i$ вычисляем НОД $d = gcf(|x_i - x_j|, n)$.

Если $n > d > 1$, то d – нетривиальный делитель n.

Если $^n/_d$ - составное число, то применяем данный алгоритм ещё раз уже к числу $^n/_d$. И так продолжаем до тех пор, пока не получим разложение только из простых чисел.

Пример использования алгоритма

Нужно разложить на простые множители число 54.

Возьмём $F = x^2 + 1$ и $x_0 = 3$.

Вычисляем последовательность по модулю 54, пока она не зациклится:

$$x_0 = 3, x_1 = 3 * 3 + 1 \ (mod\ 54) = 10,$$
$$x_2 = 47, x_3 = 50, x_4 = 17, x_5 = 20, x_6 = 23, x_7 = 44, x_8 = 47 \ldots$$

$x_8 = x_3$, значит, последовательность зациклилась.

Начинаем попарно высчитывать НОД = $gcf(\left|x_i - x_j\right|, 54)$ чисел в последовательности, пока он не будет больше 1, но меньше 54.

При $|x_3 - x_2| = 3$: НОД(3, 54) = 3. Следовательно, 3 — простой делитель 54.

54/3 = 18.

Число 18 — не простое, поэтому применяем к нему алгоритм аналогично. И так до тех пор, пока у нас не получится разложение из одних простых чисел.

Особенности использования алгоритма

Следует отметить, что рассматриваемый алгоритм в значительной степени случаен, его эффективность сильно и непредсказуемо зависит от выбора многочлена и начального элемента в последовательности.

Метод эффективен для нахождения небольших простых делителей числа n. Делители большего размера тоже могут быть обнаружены, однако лишь с некоторой вероятностью.

Тест на простоту чисел Миллера-Рабина

Для реализации факторизации чисел любого метода нужен тест на простоту чисел.

Я выбрала вероятностный тест Миллера-Рабина, так как при проверке k чисел на условия простоты вероятность того, что составное число будет принято за простое, будет меньше $({}^{1}/_{4})^{k}$. В своей программе я проверяла 3 случайных числа, так в этом случае достигается достаточно удовлетворительная вероятность 0.015625.

В моей программе функция носит название isPrimeNum().

Тест на простоту чисел Миллера-Рабина

Для проверки числа n (причём $n > 2$) на простоту, во-первых, оно представляется в виде $n - 1 = t * 2^s$, где t -нечётно.

Дальше проверяются следующие утверждения.

Если n - простое, то для любого a из Z_n $(a < n)$ выполняются:

1) $a^t = 1 \ (mod\ n)$

2) ∃ r, что $0 \le r < s: \ a^{2^s t} = -1 \ (mod\ n)$

Если хотя бы одно из этих условий соблюдается, то число a является свидетелем простоты числа n.

Делая проверку на 3 случайных числах, мы повышаем вероятность того, что число n не псевдопростое.

Возведение в степень по модулю

Данный алгоритм основывается на том факте, что для заданных a и b следующие 2 уравнения эквивалентны:

$$\mathrm{c} = a * b \ (mod \ m)$$

$$c = a * \big(\ b \ (mod \ m) \big)(mod \ m)$$

Алгоритм следующий:

- Пусть c = 1, e' = 0
- Увеличим e' на 1
- Установим $c = b * c \ (mod \ m)$
- Если $e' \ < \ e$, возвращаемся к шагу 2. В противном случае, c содержит правильный вариант ответ $\mathrm{c} = \ b^e \ (mod \ m)$.

В моей программе данный алгоритм носит название modular_pow().