Базовые алгоритмы криптографии

Алгоритмы арифметики больших чисел

Диапазон чисел, которые используются в реальных задачах криптографии, доходит до нескольких сот и даже тысячи десятичных цифр. Приведем алгоритмы арифметических операций над числами с произвольным числом разрядов (большими числами). Предполагается, что архитектура ЭВМ дает возможность выполнять элементарные арифметические операции над одно- и двухразрядными числами.

Пусть b – основание системы счисления. Целому неотрицательному числу u поставим в соответствие набор целых чисел $(u_{k-1}\, u_{k-2} \ldots u_1\, u_0)_b$ такой, что

$$u = u_{k-1}b^{k-1} + u_{k-2}b^{k-2} + \ldots + u_1 b + u_0,$$

где $0 \leq u_{k-1}, u_{k-2}, \ldots, u_1, u_0 < b$.

Алгоритм 1 (сложение). Сложение $u + v = (w_k \ldots w_1 w_0)_b$ чисел $u = (u_{k-1}\, u_{k-2} \ldots u_1\, u_0)_b$ и $v = (v_{k-1}\, v_{k-2} \ldots v_1\, v_0)_b$.

1. Установить $c \leftarrow 0$.
2. Для $i = 0, \ldots, k-1$ выполнить:

а) $w_i \leftarrow (u_i + v_i + c) \bmod b$;

б) положить $c \leftarrow 0$, если $u_i + v_i + c < b$ и $c \leftarrow 1$ в противном случае.

3. Установить $w_k \leftarrow c$.

Алгоритм 2 (вычитание). Вычитание $u - v = (w_{k-1} \ldots w_1 w_0)_b$ числа $v = (v_{k-1}\, v_{k-2} \ldots v_1\, v_0)_b$ из числа $u = (u_{k-1}\, u_{k-2} \ldots u_1\, u_0)_b$, $u \geq v$.

1. Установить $c \leftarrow 0$.
2. Для $i = 0, \ldots, k-1$ выполнить:

а) $w_i \leftarrow (u_i - v_i - c) \bmod b$;

б) положить $c \leftarrow 0$, если $u_i - v_i - c \geq 0$ и $c \leftarrow 1$ в противном случае.

Замечание. Если $u < v$, то по окончании выполнения алгоритма $c = 1$ и результат $w = u - v + b^k$.

Алгоритм 3 (умножение). Умножение $uv = (w_{k+l-1} \ldots w_1 w_0)_b$ чисел $u = (u_{k-1}\, u_{k-2} \ldots u_1\, u_0)_b$ и $v = (v_{l-1}\, v_{l-2} \ldots v_1\, v_0)_b$.

1. Для $i = 0, \ldots, k+l-1$ установить $w_i \leftarrow 0$.
2. Для $i = 0, \ldots, l-1$ выполнить:

а) $c \leftarrow 0$;

б) для $j = 0, \ldots, k-1$ вычислить $(xy)_b \leftarrow w_{i+j} + u_j v_i + c$ и установить $w_{i+j} \leftarrow y$, $c \leftarrow x$;

в) $w_{i+k} \leftarrow c$.

Алгоритм 4 (деление с остатком). Деление числа $u = (u_{k+l-1} \ldots u_1\ u_0)_b$ на число $v = (v_{k-1}\ \ldots\ v_1\ \ v_0)_b$, $k \geq 2$, $v_{k-1} \neq 0$, т.е. нахождение частного $q = (q_l \ldots q_1\ q_0)_b$ и остатка $r = (r_{k-1} \ldots r_1\ r_0)_b$ таких, что $u = qv + r$ и $0 \leq r < v$.

1. Выбрать произвольное целое число d такое, что

$$vd < b^k \text{ и}$$

$$\left[\frac{b}{2}\right] \leq v_{k-1}d < b,$$

Установить:

а) $u \leftarrow ud$, $u = (u_{k+l} \ldots u_1\ u_0)_b$;

б) $v \leftarrow vd$, $v = (v_{k-1} \ldots v_1\ v_0)_b$.

2. Для $i = k + l, k + l - 1, \ldots, k$ выполнить:

а) вычислить пробное частное

$$\tilde{q} \leftarrow \min\left(\left[\frac{u_i b + u_{i-1}}{v_{k-1}}\right], b - 1\right)$$

б) пока $\tilde{q}(v_{k-1}b + v_{k-2}) > u_i b^2 + u_{i-1}b + u_{i-2}$, выполнить $\tilde{q} \leftarrow \tilde{q} - 1$;

в) $u \leftarrow u - \tilde{q}vb^{i-k}$;

г) (*корректирующее сложение*) если $u < 0$, то установить $\tilde{q} \leftarrow \tilde{q} - 1$,

$$u \leftarrow u + vb^{i-k};$$

д) $q_{i-k} \leftarrow \tilde{q}$.

3. Установить $r \leftarrow u/d$.

Действие на шаге 1 алгоритма называется *нормализацией*. При нормализации цикл 2б выполняется не более 2 раз. В общем случае можно выбрать

$$d = \left[\frac{b}{v_{k-1} + 1}\right].$$

Проверка чисел на простоту

Проверка чисел на простоту является составной частью алгоритмов генерации простых чисел, применяемых в криптографии с открытым ключом. Алгоритмы проверки на простоту можно разделить на вероятностные и детерминированные.

Детерминированный алгоритм всегда действует по одной и той же схеме и гарантированно решает поставленную задачу (или не дает никакого ответа).

Для того, чтобы проверить вероятностным алгоритмом, является ли число n простым, выбирают случайной число a $(1 < a < n)$ и проверяют условия алгоритма. Если число n не проходит тест по основанию a, то алгоритм выдает результат «Число n составное». Если же n проходит тест по основанию a, ничего нельзя сказать о том, действительно ли число n

является простым. Последовательно проведя ряд проверок таким тестом для разных a и получив для каждого из них ответ «Число n, вероятно, простое», можно утверждать, что число n является простым с вероятностью, близкой к 1. После T независимых выполнений теста вероятность того, что составное число n будет T раз объявлено простым (вероятность ошибки), не превосходит $\frac{1}{2^T}$.

Для проверки чисел на простоту часто используется тест Миллера–Рабина.

Алгоритм 5 (тест Миллера–Рабина). Входные данные: нечетное число $n \geq 5$ и число итераций T.

1. Представить n в виде $2^s r + 1$, где s – натуральное число, r – нечетное натуральное число.

2. Для $t = 1,2, \dots, T$ выполнить:

   1) выбрать случайное целое число u, $2 \leq u \leq n - 2$;

   2) $v \leftarrow u^r \, mod \, n$;

   3) если $v = 1$ или $v = n - 1$, то перейти к 2.6;

   4) для $i = 1,2, \dots, s - 1$ выполнить:

      (a) $v \leftarrow v^2 \, mod \, n$;

      (b) если $v = 1$, то возвратить "Число n составное";

      (c) если $v = n - 1$, то перейти к шагу 2.6;

   5) возвратить "Число n составное";

   6) продолжить.

3 Возвратить "Число n, вероятно, простое".

Лабораторная работа 2

Цель работы: изучить алгоритмы выполнения арифметических операций над числами с произвольным числом разрядов (большими числами) и тест Миллера–Рабина для проверки чисел на простоту.

Задания. 1. Разработайте класс «BigNumberOperations». Реализуйте в данном классе методы для выполнения арифметических операций над большими числами. Доработайте алгоритм вычитания для случая, когда результат операции является отрицательным.

2. Реализуйте алгоритм 5 (тест Миллера–Рабина) для проверки чисел на простоту.

Входные данные: нечетное число $n \geq 5$ и число итераций T.

Выходные данные: представление числа n в виде $2^s r + 1$,
значения чисел u и v на каждой итерации,
ответ "Число n составное" или "Число n, вероятно, простое".