НАО «Восточно-Казахстанский технический университет имени Д. Серикбаева»

Дисциплина «Теоретико-числовые методы в криптографии»

ПРАКТИЧЕСКАЯ РАБОТА 9

СРАВНЕНИЯ И КИТАЙСКАЯ ТЕОРЕМА ОБ ОСТАТКАХ

Методические указания и задание к выполнению

Тема 9. Аудиторная нагрузка: 2 академических часа. Перед выполнением заданий изучите методичку и разобранные примеры. Программные фрагменты можно подготовить заранее и использовать для проверки ручных решений.

1 Цель и результаты работы

Научиться решать сравнения первого порядка, определять число решений и объединять условия по разным модулям. После работы студент должен уметь проверять совместность системы, находить её полный период и восстанавливать целое число по остаткам.

Понадобятся алгоритм Евклида, тождество Безу и операции с вычетами. Для кода достаточно Python ≥3.9 без дополнительных пакетов. Все переменные в программах — целые числа; модуль одиночного сравнения не меньше 2.

2 Методические сведения

2.1 Сравнение первого порядка

Сравнение $ax \equiv b \pmod m$ означает, что m делит $ax - b$. Искомый объект — класс целых чисел, а для перечисления решений обычно выбираются представители $0 \le x < m$.

Положим $d = \gcd(a, m)$. Сравнение разрешимо тогда и только тогда, когда $d \mid b$. Если это условие выполнено, имеется ровно d различных решений по модулю m.

Обоснование. Из $ax - b = km$ следует $d \mid b$. Обратно, при $d \mid b$ делим все три числа на d:

$$\frac{a}{d}x \equiv \frac{b}{d} \pmod{m/d}.$$

Теперь $\gcd(a/d, m/d) = 1$, поэтому при $m/d > 1$ коэффициент имеет обратный элемент. Найденное решение x_0 по модулю m/d даёт все представители по исходному модулю:

$$x = x_0 + k\frac{m}{d}, \qquad k = 0,1,\dots,d-1.$$

Если $m/d = 1$, то исходное сравнение при $d \mid b$ выполняется для любого целого x. Его представители — все числа от 0 до $m - 1$. Для $a = 0$ это даёт два случая: все классы при $m \mid b$ и отсутствие решений в противном случае.

Правило сокращения. Общий делитель необходимо сокращать одновременно в коэффициенте, правой части и модуле. Например, $2x \equiv 2 \pmod 6$ превращается в $x \equiv 1 \pmod 3$, а не в $x \equiv 1 \pmod 6$. По модулю 6 имеются два решения: 1 и 4.

2.2 Обратный элемент и расширенный алгоритм Евклида

Если $\gcd(a, m) = 1$, найдём целые u, v из равенства $au + mv = 1$. Тогда u является обратным к a по модулю m, а решение имеет вид $x \equiv ub \pmod m$.

Пример 1. Решим $14x \equiv 30 \pmod{100}$. Здесь $d = 2$ делит 30. После сокращения получаем $7x \equiv 15 \pmod{50}$. Алгоритм Евклида даёт $50 = 7 \cdot 7 + 1$, откуда $1 = 50 - 7 \cdot 7$. Следовательно, $7^{-1} \equiv -7 \equiv 43 \pmod{50}$ и

$$x_0 \equiv 43 \cdot 15 \equiv 45 \pmod{50}.$$

По модулю 100 ответ равен $x \in \{45, 95\}$. Проверка: $14 \cdot 45 - 30 = 600$, $14 \cdot 95 - 30 = 1300$; обе разности делятся на 100.

Пример 2. Сравнение $12x \equiv 7 \pmod{18}$ решений не имеет, поскольку $\gcd(12, 18) = 6$, а 7 не делится на 6. Искать обратный к 12 по модулю 18 нельзя.

Пример 3. В сравнении $-3x \equiv 6 \pmod{15}$ можно заменить коэффициент на 12, но это необязательно. После деления на 3 получаем $-x \equiv 2 \pmod{5}$, то есть $x \equiv 3 \pmod{5}$. Представители по модулю 15 равны 3,8,13.

2.3 Китайская теорема об остатках

Теорема. Если положительные модули $m_1, \dots, m_k$ попарно взаимно просты, то система $x \equiv r_i \pmod{m_i}$ имеет единственное решение по модулю $M = m_1 \cdots m_k$.

Для вычисления положим $M_i = M/m_i$ и найдём $u_i \equiv M_i^{-1} \pmod{m_i}$. Тогда

$$x \equiv \sum_{i=1}^{k} r_i \, M_i u_i \pmod{M}.$$

Почему формула работает. По модулю m_i слагаемое $r_i M_i u_i$ равно r_i, а все остальные содержат множитель m_i и обращаются в нуль. Если x и y удовлетворяют системе, их разность делится на каждый m_i. Попарная взаимная простота означает, что разность делится на произведение M, чем доказана единственность класса.

Пример 4. Решим систему

$$x \equiv 2 \pmod{3}, \qquad x \equiv 3 \pmod{5}, \qquad x \equiv 2 \pmod{7}.$$

Здесь $M = 105$, а $M_1 = 35$, $M_2 = 21$, $M_3 = 15$. Обратные элементы равны соответственно 2,1,1. Поэтому

$$x \equiv 2 \cdot 35 \cdot 2 + 3 \cdot 21 + 2 \cdot 15 = 233 \equiv 23 \pmod{105}.$$

Все целые решения: $x = 23 + 105t$, $t \in \mathbb{Z}$. Проверка остатков: $23 \bmod 3 = 2$, $23 \bmod 5 = 3$, $23 \bmod 7 = 2$. Если дополнительно требуется $0 \le x < 105$, выбирается единственное число 23. Без ограничения на интервал система определяет класс, а не одно целое число.

2.4 Системы с общими делителями модулей

Для системы $x \equiv r \pmod{m}$, $x \equiv s \pmod{n}$ положим $d = \gcd(m, n)$. Решение существует тогда и только тогда, когда

$$s - r \equiv 0 \pmod{d}.$$

При совместности все решения образуют один класс по модулю $L = \operatorname{lcm}(m, n) = mn/d$.

Вывод алгоритма. Из первого условия $x = r + mt$. Подстановка во второе даёт $mt \equiv s - r \pmod{n}$. По критерию линейного сравнения оно разрешимо ровно при $d \mid (s - r)$. Сократив на d, находим один класс t по модулю n/d, поскольку m/d и n/d взаимно просты. Изменение t на n/d меняет x на mn/d, откуда следует период L.

Пример 5. Пусть $x \equiv 5 \pmod{12}$ и $x \equiv 11 \pmod{18}$. Имеем $d = 6$ и $11 - 5 = 6$, поэтому система совместна. Полагаем $x = 5 + 12t$:

$$12t \equiv 6 \pmod{18}, \qquad 2t \equiv 1 \pmod{3}, \qquad t \equiv 2 \pmod{3}.$$

Отсюда $x \equiv 29 \pmod{36}$. По модулю произведения 216 было бы шесть представителей 29,65,101,137,173,209, но минимальный положительный период множества решений равен 36.

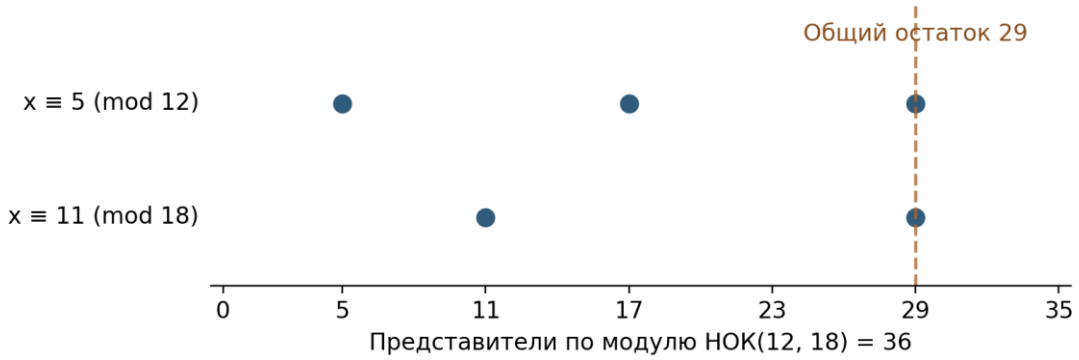


Рисунок 1. На интервале от 0 до 35 обеим сравнениям удовлетворяет только число 29.

Пример 6. Если второе условие заменить на $x \equiv 10 \pmod{18}$, разность $10 - 5 = 5$ не делится на 6. Система несовместна. Использовать формулу КТО с обратным к 12 по модулю 18 здесь нельзя.

Систему из нескольких сравнений можно решать последовательным объединением: после каждого шага хранить найденный остаток и период. Если на некотором шаге возникает несовместность, исходная система не имеет решений. Если все шаги успешны, итоговый период равен НОК исходных модулей.

2.5 Нормализация системы линейных сравнений

Если система записана в виде $a_i x \equiv b_i \pmod{m_i}$, сначала проверяют каждое сравнение отдельно. Разрешимое сравнение сводится к одному классу по сокращённому модулю $m_i/\gcd(a_i, m_i)$. После этого применяется алгоритм объединения классов. Нельзя произвольно выбирать один из представителей по исходному модулю и отбрасывать остальные.

Пример 7. Решим $6x \equiv 8 \pmod{14}$ и $4x \equiv 4 \pmod{12}$. Первое сравнение даёт $3x \equiv 4 \pmod{7}$, откуда $x \equiv 6 \pmod{7}$. Второе даёт $x \equiv 1 \pmod{3}$. Полагаем $x = 6 + 7t$ и находим $t \equiv 1 \pmod{3}$. Ответ: $x \equiv 13 \pmod{21}$. Подстановка даёт $6 \cdot 13 - 8 = 70$ и $4 \cdot 13 - 4 = 48$.

2.6 Учебное восстановление сообщения RSA

КТО позволяет восстанавливать результат по остаткам. Возьмём маленькие учебные параметры $p = 11, q = 13, n = 143, \varphi(n) = 120, e = 7$. Из $7 \cdot 103 = 1 + 6 \cdot 120$ получаем закрытый показатель $d = 103$.

Для сообщения $M = 42$ шифротекст равен $c \equiv 42^7 \equiv 81 \pmod{143}$. Найдём остатки результата расшифрования:

$$m_p \equiv 81^3 \equiv 9 \pmod{11}, \qquad m_q \equiv 81^7 \equiv 3 \pmod{13}.$$

Показатели 3 и 7 — остатки d по модулям 10 и 12. Если шифротекст делится на соответствующее простое число, остаток расшифрования сразу равен 0; в этом примере оба основания ненулевые. При данных параметрах сокращённые показатели положительны и для нулевых оснований также дают верный результат.

Восстановим число: $M = 9 + 11t$, $11t \equiv -6 \equiv 7 \pmod{13}$. Поскольку $11^{-1} \equiv 6 \pmod{13}$, имеем $t \equiv 3 \pmod{13}$, откуда $M \equiv 42 \pmod{143}$. В диапазоне $0 \le M < 143$ получаем исходное сообщение 42. Это арифметический учебный пример; описания полноценной схемы шифрования он не заменяет.

3 Примеры программ перед выполнением заданий

Сохраните следующие блоки в один файл в приведённом порядке. Функция `linear` возвращает все представители по исходному модулю. Функции `merge` и `crt` возвращают пару «наименьший неотрицательный остаток, период» либо `None`, если система несовместна.

3.1 Тождество Безу и обратный элемент

На каждом шаге расширенного алгоритма Евклида остатки представлены линейными комбинациями исходных чисел. Одновременное обновление коэффициентов сохраняет это равенство. В конце возвращаются g, u, v с $au + bv = g = \gcd(a, b) \geq 0$.

```
def egcd(a, b):
    r0, r1 = a, b
    s0, s1, t0, t1 = 1, 0, 0, 1
    while r1:
        q = r0 // r1
        r0, r1 = r1, r0 - q * r1
        s0, s1 = s1, s0 - q * s1
        t0, t1 = t1, t0 - q * t1
    if r0 < 0:
        r0, s0, t0 = -r0, -s0, -t0
    return r0, s0, t0


def inverse(a, m):
    if m < 2:
        raise ValueError("Модуль должен быть не меньше 2")
    g, u, _ = egcd(a, m)
    if g != 1:
        raise ValueError("Обратного элемента нет")
    return u % m
```

3.2 Все решения линейного сравнения

Коэффициент u из $au + mv = g$ сразу даёт частное решение после умножения на b/g. Остальные представители отличаются на m/g. Формула работает и при $a = 0$, если условие разрешимости выполнено.

```
def linear(a, b, m):
    if m < 2:
        raise ValueError("Модуль должен быть не меньше 2")
    g, u, _ = egcd(a, m)
    if b % g:
        return []
    step = m // g
    x0 = (u * (b // g)) % step
    return [x0 + k * step for k in range(g)]
```

3.3 Объединение двух сравнений

Здесь модуль 1 разрешён как служебный случай: условие по модулю 1 не ограничивает целое число. Остатки сначала нормализуются, поэтому отрицательные входные значения допустимы. В формуле не требуется вычислять обратный по модулю 1.

```
def merge(r, m, s, n):
    if m < 1 or n < 1:
        raise ValueError("Модули должны быть положительными")
    r, s = r % m, s % n
    g, u, _ = egcd(m, n)
    if (s - r) % g:
        return None
    step = n // g
    t = ((s - r) // g * u) % step
    period = m * step
    return (r + m * t) % period, period
```

3.4 Объединение всей системы

Начальная пара `(0, 1)` описывает все целые числа. На каждом шаге к уже учтённым условиям добавляется одно новое. Программа применима и к попарно взаимно простым модулям, и к общему случаю.

```
def crt(residues, moduli):
    if len(residues) != len(moduli) or not moduli:
        raise ValueError("Нужны непустые списки одной длины")
    result = (0, 1)
    for s, n in zip(residues, moduli):
        result = merge(result[0], result[1], s, n)
        if result is None:
            return None
    return result
```

3.5 Запуск разобранных примеров

Вызов `pow(a, k, m)` вычисляет остаток степени без предварительного построения огромного целого числа a^k. Он применяется в учебном примере RSA.

```
print("Линейное:", linear(14, 30, 100))
print("Нет решений:", linear(12, 7, 18))
print("КТО:", crt([2, 3, 2], [3, 5, 7]))
print("Общие модули:", merge(5, 12, 11, 18))
print("Несовместность:", merge(5, 12, 10, 18))
c, d = 81, 103
mp, mq = pow(c, d % 10, 11), pow(c, d % 12, 13)
print("RSA:", mp, mq, crt([mp, mq], [11, 13]))
```

Ожидаемый вывод

```
Линейное: [45, 95]
Нет решений: []
КТО: (23, 105)
Общие модули: (29, 36)
Несовместность: None
RSA: 9 3 (42, 143)
```

3.6 Самопроверка и границы применимости

В проверках присутствуют нулевой и отрицательный коэффициенты, несовместность и вложенные модули. Сравнение с перебором допустимо для малых учебных модулей и независимо подтверждает полноту списка решений.

```
from math import lcm

assert linear(0, 0, 5) == [0, 1, 2, 3, 4]
assert linear(0, 1, 5) == []
assert linear(-3, 6, 15) == [3, 8, 13]
assert crt([-1, 2], [5, 7]) == (9, 35)
assert merge(2, 6, 2, 3) == (2, 6)
assert merge(1, 2, 0, 4) is None
for a, b, m in [(14, 30, 100), (12, 7, 18), (9, 0, 27)]:
    expected = [x for x in range(m) if (a*x-b) % m == 0]
    assert linear(a, b, m) == expected
r, period = crt([2, 3, 2], [3, 5, 7])
assert period == lcm(3, 5, 7)
assert all(r % m == a for a, m in zip([2, 3, 2], [3, 5, 7]))
print("Проверки пройдены")
```

Ожидаемая последняя строка: `Проверки пройдены`. Все проверки должны выполняться при обычном запуске Python без отключения операторов `assert`.

Алгоритм Евклида использует логарифмическое число делений по величине положительных входов. Функция `linear` дополнительно создаёт список из g ответов, поэтому требует $O(g)$ памяти и времени на его заполнение. Для больших g лучше хранить решение как класс по модулю m/g. В `crt` выполняется одно объединение на условие; при оценке общей сложности нужно учитывать рост промежуточного периода и стоимость операций над большими целыми числами.

4 Порядок выполнения работы

1. Разобрать примеры 1–7 и запустить код методички до занятия либо в начале работы.
2. Выписать данные своего варианта из двух таблиц раздела 6. Номер строки в обеих таблицах должен совпадать.
3. Выполнить задания 1–4 вручную с объяснением условий разрешимости. Затем проверить ответы программой.
4. Выполнить учебное восстановление по КТО в задании 5 и сопоставить с прямым возведением в степень.
5. Оформить отчёт и ответить на контрольные вопросы преподавателя.

Рекомендуемое распределение 90 минут: обсуждение методики — 10; задание 1 — 15; задание 2 — 20; задания 3–4 — 20; задание 5 — 15; проверка и выводы — 10. Программные определения даются готовыми, их переписывание с нуля не входит в аудиторную нагрузку.

5 Индивидуальные задания

Задание 1. Линейное сравнение. Для своей строки таблицы 1 решить $ax \equiv b \pmod{m}$. Найти НОД, проверить условие разрешимости и выписать все различные представители $0 \le x < m$. Если решений нет, обосновать это делимостью. Для разрешимого случая показать тождество Безу и сокращённый модуль.

Задание 2. КТО для трёх модулей. Решить систему $x \equiv r_i \pmod{m_i}$ по данным таблицы 1. Проверить попарную взаимную простоту, вычислить M, M_i, u_i и применить формулу суммы. Выписать все целые решения и единственного представителя в диапазоне $0 \le x < M$. Проверить три остатка.

Задание 3. Общие делители модулей. По таблице 2 решить систему $x \equiv s_1 \pmod{n_1}$, $x \equiv s_2 \pmod{n_2}$. Указать НОД, разность остатков и условие совместности. При совместности найти наименьший неотрицательный остаток и минимальный положительный период. Объяснить, почему период равен НОК, а не обязательно произведению.

Задание 4. Анализ изменения данных. Заменить s_2 из задания 3 на $s_2 + 1$, сохранив остальные данные. Определить совместность изменённой системы без полного решения. Отдельно решить общую для всех систему $6x \equiv 8 \pmod{14}$, $4x \equiv 4 \pmod{12}$ и объяснить сокращение модулей.

Задание 5. Учебный пример RSA и КТО. Для номера варианта j положить $M = 10 + j$. При $n = 143$, $e = 7$, $d = 103$ вычислить $c = M^e \bmod n$, остатки $m_p = c^3 \bmod 11$ и $m_q = c^7 \bmod 13$. Восстановить сообщение по КТО и проверить равенство с $c^{103} \bmod 143$. Для степеней разрешается использовать `pow`; восстановление по КТО показать вручную. Объяснить роль ограничения $0 \le M < 143$.

6 Варианты индивидуальных заданий

Таблица 1. Линейное сравнение и система с попарно взаимно простыми модулями. В тройках остатки и модули записаны в одинаковом порядке.

№	a	b	m	(m_1, m_2, m_3)	(r_1, r_2, r_3)
1	3	9	13	(5,7,8)	(1,5,2)
2	8	2	30	(7,9,10)	(1,7,3)
3	15	24	51	(8,9,11)	(4,6,5)
4	24	68	76	(5,11,12)	(2,0,5)
5	35	26	55	(5,7,8)	(4,3,6)
6	-8	1	13	(7,9,10)	(6,3,1)
7	4	6	30	(8,9,11)	(0,2,7)
8	9	39	51	(5,11,12)	(0,2,1)
9	16	24	76	(5,7,8)	(2,1,2)
10	25	26	55	(7,9,10)	(4,8,9)
11	6	0	13	(8,9,11)	(4,7,9)
12	-14	14	30	(5,11,12)	(3,4,9)
13	24	3	51	(5,7,8)	(0,6,6)
14	8	52	76	(7,9,10)	(2,4,7)
15	15	36	55	(8,9,11)	(0,3,0)

№	a	b	m	(m_1, m_2, m_3)	(r_1, r_2, r_3)
16	4	7	13	(5,11,12)	(1,6,5)
17	14	26	30	(5,7,8)	(3,4,2)
18	-18	48	51	(7,9,10)	(0,0,5)
19	28	56	76	(8,9,11)	(4,8,2)
20	40	1	55	(5,11,12)	(4,8,1)

Таблица 2. Система с общими делителями модулей для заданий 3 и 4.

№	n_1	n_2	s_1	s_2
1	12	15	4	1
2	16	20	9	5
3	20	25	14	9
4	8	10	3	4
5	12	15	4	7
6	16	20	13	1
7	20	25	10	20
8	8	10	7	0
9	12	15	4	13
10	16	20	1	17
11	20	25	6	6
12	8	10	3	6
13	12	15	4	4
14	16	20	5	13
15	20	25	2	17
16	8	10	7	2
17	12	15	4	10
18	16	20	9	9
19	20	25	18	3
20	8	10	3	8

7 Требования к отчёту

Указать тему, ФИО, группу и номер варианта. Привести исходные данные, условия применимости каждого метода, ручные вычисления и полные ответы с модулями. Добавить использованный код и результаты проверки. Скриншот запуска не заменяет доказательства числа решений или несовместности системы.

В выводе объяснить различие между представителем, классом решений и периодом. Указать, в каких задачах применялась обычная КТО, а в каких потребовалось учитывать общий делитель модулей.

8 Критерии оценивания

Составляющая	Баллы
Линейное сравнение и полнота ответа	20
КТО для трёх модулей с проверкой	25
Общий случай двух модулей	20
Изменение данных и нормализация системы	15
Восстановление RSA по КТО	15
Оформление и программная проверка	5
Всего	100

9 Контрольные вопросы

1. Что означает сравнение $ax \equiv b \pmod{m}$?
2. Как НОД определяет существование и число решений?
3. Почему при сокращении сравнения меняется модуль?
4. Как тождество Безу даёт обратный элемент?
5. Каковы условия обычной китайской теоремы об остатках?
6. Почему формула суммы удовлетворяет каждому сравнению?
7. По какому модулю решение КТО единственно?
8. Как проверить совместность двух условий при невзаимно простых модулях?
9. Почему в общем случае период равен НОК?
10. Как обрабатывать систему линейных сравнений с коэффициентами?
11. Что возвращают функции `linear`, `merge` и `crt`?
12. Как КТО применяется при учебном восстановлении RSA?

Литература

1. Мартынов Л. М. Алгебра и теория чисел для криптографии. 2020. Разделы о сравнениях и системах сравнений.
2. Trappe W., Washington L. C. Introduction to Cryptography with Coding Theory. 2nd ed. 2006. Глава 3 Basic Number Theory, раздел о китайской теореме об остатках.
3. Paar C., Pelzl J. Understanding Cryptography. 2010. §7.5.2 Fast Decryption with the Chinese Remainder Theorem.