4.7 Подведение итогов главы

Теория чисел. Простые числа, остатки от деления, сравнения по модулю — основы модулярной арифметики. Алгоритм Евклида и его расширенная версия позволяют не только находить НОД, но и решать модулярные линейные уравнения. Быстрое возведение в степень за $O(\log k)$ операций позволяет работать с гигантскими числами; та же идея даёт логарифмический алгоритм для чисел Фибоначчи через возведение матрицы 2×2 в степень. Малая теорема Ферма $a^{p-1} \equiv 1 \pmod{p}$ — ключевое утверждение, обеспечивающее корректность RSA. Простых чисел много (закон $\pi(N) \sim N / \ln N$), и проверка простоты быстра (тест Миллера–Рабина, AKS), но *факторизация* составного числа на множители — задача экспоненциальной сложности.

История криптографии. От шифра Цезаря — к симметричным шифрам с одним общим ключом — к проблеме доставки ключа. Идея **односторонней функции** (легко в одну сторону, очень сложно в обратную) и **функции с секретом** (трапдор) даёт возможность строить асимметричные шифры с открытым и закрытым ключами.

RSA и Диффи–Хеллман. RSA опирается на сложность факторизации $n = pq$: открытый ключ (n, e) позволяет шифровать, закрытый $d = e^{-1} \bmod \varphi(n)$ — расшифровывать. Корректность доказывается малой теоремой Ферма. Диффи–Хеллман — протокол выработки общего ключа по открытому каналу: его стойкость основана на сложности задачи дискретного логарифма. Обе схемы уязвимы к атаке «человек посередине» без аутентификации.

Применения RSA. Та же конструкция, применённая в разных порядках, решает четыре задачи: шифрование сообщения, аутентификация пользователя через challenge-response, электронная цифровая подпись (подпись закрытым ключом, проверка открытым) и слепая подпись (основа протоколов электронного голосования).

Хеширование. Универсальное семейство Картера–Вегмана $h_a(x) = (a_1 x_1 + \ldots + a_k x_k) \bmod p$ (p — простое) гарантирует, что вероятность коллизии любых двух ключей при случайном выборе a не превосходит $1/p$. Ключевую роль играет существование и единственность обратного элемента по простому модулю — результат расширенного алгоритма Евклида.

Вероятностные счётчики. HyperLogLog оценивает число различных элементов в потоке с точностью $\sim 1\%$, используя всего килобайты

памяти. В сочетании с теорией графов это позволяет современным сетям подтверждать гипотезу Стенли Мильграма о малом мире: средняя длина «социальной» цепочки в крупных сетях — около *четырёх рукопожатий*.

Что почитать дальше

- *Музыкантский А. И., Фурин В. В.* Лекции по криптографии. М.: МЦНМО, разные годы. — Доступное изложение криптографических протоколов, в том числе электронного голосования; рекомендуется к разделу 1.3 и главе 2.
- *Литвак Н. В., Райгородский А. М.* Кому нужна математика? Содержательное введение в математику и computer science. — Главы 6, 7 и приложения к ним; материал о малых мирах, графах и вероятностных алгоритмах.
- *Дасгупта С., Пападимитриу Х., Вазирани У.* Алгоритмы. — Глава о хешировании, главы об алгоритмах теории чисел.
- *Кнут Д.* Искусство программирования, том 2 (Получисленные алгоритмы). — Классический фундаментальный труд по алгоритмам теории чисел.
- *Виноградов И. М.* Основы теории чисел. — Классический российский учебник теории чисел, доступный школьникам старших классов.

Большой итоговый проект

В качестве длинного исследовательского проекта на каникулы или на год предлагаем выбрать одно из следующих направлений:

- **«Своя» реализация RSA.** На любом удобном языке программирования (Python, Java, C++) реализуйте полный цикл: генерацию пары ключей (e, n) и d (с честным алгоритмом Миллера–Рабина для подбора простых), шифрование, расшифрование, цифровую подпись. Протестируйте на примерах из этой главы.
- **Атака на «слабый» RSA.** Возьмите малое n (50–100 битов) и попытайтесь его факторизовать. Сравните скорость наивного перебора и метода Полларда ρ. Подумайте, при каких размерах n ваш ноутбук «сдаётся».

- **Симуляция «малого мира».** Сгенерируйте случайный граф из 1000–10000 вершин по модели Уоттса–Строгаца. Вычислите средние длины кратчайших путей. Сравните с предсказанием теории.
- **Свой HyperLogLog.** Реализуйте упрощённый вариант HyperLogLog ($b = 8$ или 10). Сгенерируйте поток из миллиона случайных идентификаторов с заранее известным числом уникальных. Оцените точность алгоритма.
- **Стойкость хеш-функции.** Возьмите криптографическую хеш-функцию (например, SHA-1 уже взломан) и реализуйте поиск коллизий простым перебором. Сравните с теоретической оценкой «парадокса дней рождения».