Лекция 3

ЗАДАЧИ КЛАССОВ P И NP

Классическая криптография разрабатывает хитроумные ключи, которые защищают информацию от несанкционированного доступа. Современная криптография основана на другой, весьма остроумной, идее, которая заключается в том, что информацию можно защитить не только скрывая ключи к ней, но и используя свойство *трудоемкости* или *сложности* вычислений.

Это свойство означает, что алгоритм дешифрования (даже будучи известным всем сторонам) может быть настолько трудоемким, что его реализация невозможна даже с использованием современных вычислительных возможностей. Вместе с тем, знание небольшого количества параметров (скрытых от непосвященных сторон) позволяет достаточно просто реализовать этот алгоритм.

Эту идею мы продемонстрируем на следующем довольно простом примере.

1. Как записать последний ход в шахматной партии?

`Алиса` и `Боб` играют в шахматы по телефону. Они столкнулись с необходимостью прервать игру на ночь. Последний ход делает `Алиса`. Как сделать так, чтобы `Боб`, зная ход `Алисы`, не имел преимущества перед `Алисой`, размышляя над своим ответом целую ночь? Очевидно, что для этого последний ход `Алисы` надо скрыть от `Боба`. Но как тогда сделать так, чтобы `Алиса`, размышляя всю ночь над своим последним ходом, не изменила утром своего вечернего решения?

В настоящих шахматных турнирах также иногда возникает необходимость отложить партию на следующий день. В таких случаях ход на шахматной доске не делается, а записывается на листе бумаги, который вкладывается в конверт, а конверт запечатывается судьей. На следующий день конверт открывается и только в тот момент противник узнает о ходе.

Однако в нашей истории нет судьи, нет конверта, а обмен информацией производится по телефону. При откладывании партии `Алиса` должна сообщить некую информацию `Бобу`. При возобновлении игры она должна сообщить некую дополнительную информацию, которую будем называть *ключом*. Ключ позволит `Бобу` однозначно восстановить последний ход `Алисы`. При этом `Боб` не должен иметь возможности раскодировать ход без ключа, но в то же время он должен быть уверен, что информации, полученной вчера, достаточно чтобы помешать `Алисе` изменить свое решение в течение ночи.

Все поставленные условия, как может показаться, вместе не могут быть выполнены. Действительно, если при откладывании партии `Алиса` предоставляет достаточно информации о своем ходе, то `Боб` сможет быстро его восстановить и тогда получит преимущество. Если же информация `Алисы` допускает несколько расшифровок, то тогда уже она имеет преимущество, поскольку ночью она может выбрать оптимальный ход, а утром сообщить `Бобу` такой ключ, который укажет именно на этот оптимальный ход.

Действительно, в рамках классической теории решить эту дилемму невозможно. Но на помощь приходит принцип *сложности* вычислений: просто “*иметь*” информацию недостаточно; надо “*уметь*” ее обработать.

Используя отмеченный принцип, одно из решений описанной задачи легко получается с помощью элементарной теории чисел. Прежде всего, `Алиса` и `Боб` договаривают-

ся об использовании цифровой нотации шахматных ходов. Например, ход конем на поле `f3` можно записать `Kf3`. Если использовать порядковый номер букв в алфавите, то этот же ход можно записать в виде `1163` (“`К`” — одиннадцатая буква русского алфавита, “`f`” — шестая буква латинского). До сих пор мы всего лишь ввели обозначения.

Однако далее начинается нечто необычное. `Алиса`

(1) находит 200-значное простое число, которое начинается с последовательности `1163`; обозначим его p;
(2) находит 201-значное простое число; обозначим его q;
(3) вычисляет произведение $N = pq$; N — это число, имеющее 400 или 401 десятичных цифр.

Последнее действие невозможно осуществить вручную, но с момощью компьютера его можно выполнить в считанные мгновения. Теперь `Алиса`

(4) сообщает `Бобу` число N.

Число имеет N два делителя, p и q, но `Боб` их не знает. На следующий день `Алиса`

(5) сообщает `Бобу` числа p и q.

`Боб` легко восстанавливает ход `Алисы` по четырем первым цифрам меньшего из чисел p и q. Чтобы убедиться, что `Алиса` не обманывает его, `Боб` вычисляет произведение pq.

Несложно проверить, что описанная процедура действительно решает поставленную задачу. Прежде всего, `Алиса` не может изменить свое решение в течение ночи, поскольку число N содержит всю информацию о ее ходе: ход записан в первых четырех цифрах меньшего из двух делителей N.

Рис. 1. Шахматы по телефону

Все же сомнения остаются. На первый взгляд кажется, что Боб быстро раскроет тайну Алисы: для этого ему достаточно разложить N на множители (эта задача называется *факторизацией*), но это ему не под силу, так как N состоит из 400 (или даже больше) десятичных цифр. Факторизация такого числа займет тысячи лет даже с использованием супермощных современных компьютеров.

А может ли Алиса обмануть Боба, послав ему другую пару простых чисел вместо (p, q)? Нет, это невозможно согласно теореме о единственности разложения натурального числа на простые сомножители.

Важнейшей особенностью описанного решения является *вычислительная сложность* задачи факторизации натуральных чисел.

2. Существуют ли такие простые числа?

В описанном примере, как и во всех других настоящих применениях современной криптографии, используются супербольшие простые числа. Известно, что простых чисел бесконечно много. Но существуют ли 200-значные простые числа, которые начинаются, например, с 1163? Да, существуют. Первое из таких чисел можно найти с помощью

пакетов научных вычислений *Maple* или *Mathematica* за 1–2 секунды. Это число имеет любопытную структуру

$$1163\,\underbrace{0000000000000000000\ldots0000000000000000000}_{193\text{ нулей}}\,371$$

Впрочем, использование этого числа в качестве p было бы неразумным для `Алисы`, так как `Боб` обязательно начал бы свои поиски именно с него.

Существуют ли другие 200-значные простые числа, начинающиеся с 1163? Да, причем так много, что никому не под силу перебрать даже малую их часть. Чтобы оценить количество таких чисел, воспользуемся теоремой об асимптотике простых чисел (см. формулу (1.12)):

$$\pi(x) \sim \frac{x}{\log x}, \qquad x \to \infty.$$

Здесь $\pi(x)$ обозначает количество простых чисел, не превосходящих x. Мы хотим оценить количество простых чисел p, удовлетворяющих ограничениям

$$1163\,\underbrace{000\ldots000}_{196\text{ нулей}} \le p \le 1163\,\underbrace{999\ldots999}_{196\text{ девяток}}.$$

С небольшой потерей строгости можно считать, что для больших x

$$\pi(x) \approx \frac{x}{\log x}.$$

Эта аппроксимация для наших простых рассуждений означает, что 200-значных простых чисел, начинающихся с 1163,

существует приблизительно

$$\begin{aligned}\pi(1164\times 10^{196}-1)-\pi(1163\times 10^{196})&\\ \approx \frac{1164\cdot 10^{196}-1}{\ln(1164\cdot 10^{196}-1)}-\frac{1163\cdot 10^{196}}{\ln(1163\cdot 10^{196})}&\\ \approx \frac{1164\cdot 10^{196}-1163\cdot 10^{196}}{\ln\left(1164\cdot 10^{196}\right)}\approx \frac{10^{196}}{\ln\left(1164\right)+196\ln(10)}&\\ \approx \frac{10^{196}}{7.06+196\cdot 2.30}\approx 1.95\times 10^{193}.&\end{aligned}$$

Осознать астрономическую величину числа в правой части формулы поможет напоминание о том, что количество атомов во всей Вселенной не превосходит 10^{77}.[1]

3. Как выбрать подходящее простое число?

Чтобы обеспечить сохранность своего хода, `Алиса` может применить второй принцип современной компьютерной математики, называемый *рандомизацией*. А именно, `Алиса` может сгенерировать подходящее p с использованием датчика случайных чисел. Поскольку первые четыре цифры p зарезервированы за комбинацией 1163, то каждую из остальных 196 цифр `Алиса` может выбрать случайно. Первая попытка не обязательно приведет к успеху: получившееся 200-значное число не обязательно будет простым. В этом случае `Алисе` надо будет еще раз сгенерировать 196 случайных цифр. Но и эта попытка может оказаться неудачной. Есть ли надежда получить таким образом простое число за какое-либо разумное время?

[1] Полезно также заметить, что 10^{80} в тысячу раз больше, чем 10^{77}. Во сколько же раз 10^{193} больше, чем количество всех атомов в нашей Вселенной?

Из основной теоремы о простых числах можно, в частности, заключить, что простых 200-значных чисел примерно в 460 раз меньше, чем всех 200-значных чисел:

$$\frac{x}{\pi(x)} \approx \ln(x) \approx \ln\left(10^{200}\right) \approx 460.$$

Это означает, что примерно за 460 попыток генерации 200-значных чисел `Алиса` получит простое число. Конечно, подобные вычисления вручную осуществить невозможно, но современные компьютеры выполнят их за пару секунд.

4. Является ли число простым?

`Алисе` придется много раз проверять сгенерированные ею числа на простоту. Традиционные алгоритмы проверки не работают эффективно для 200-значных чисел. Например, для проверки является ли число n простым, можно последовательно делить его на все числа, меньшие $\sqrt{n}$. Для 200-значных n это будет означать проверку делением на все числа до $\sqrt{10^{200}} = 10^{100}$. Это работа на миллионы лет всем суперкомпьютерам мира. Как же это может сделать `Алиса`? Ей еще раз поможет принцип рандомизации.

4.1. Рандомизированный алгоритм проверки на простоту. На практике, проверка чисел на простоту больших чисел осуществляется алгоритмами, которые не являются безупречно строгими. Это означает, что такие алгоритмы дают правильный ответ лишь с определенной вероятностью.

Как правило, вероятностью правильного ответа подобного алгоритма “заведует” специальный параметр. Повторяя алгоритм достаточное количество раз с разными (случайными) значениями параметра, можно добиться какой угодно степени вероятностной уверенности, если проверяемый кандидат каждый раз проходит испытания алгоритмом.

Наиболее популярным методом описанного типа является *тест Миллера–Рабина*, основанный на малой теореме Ферма. В очень упрощенной форме одна итерация этого метода состоит в следующем.

Input: Натуральное число n.
Output: Ответ на вопрос: "*простое ли n?*":
$s = 0$, если n составное;
$s = 1$, если существует шанс, что n простое.
1. Случайным образом выбрать число a.
2. Вычислить $y = (a^n - a) \bmod n$.
3. Если $y = 1$, то $s = 1$; иначе $s = 0$.
4. Закончить алгоритм.

Алгоритм 1. Одна итерация алгоритма Миллера–Рабина

Если этот алгоритм провести многократно и каждый раз получить ответ " ... *имеется шанс, что n простое число* ... ", то уверенность в простоте n возрастает с каждой последующей итерацией. При 20 кратном повторении вероятность неправильного ответа не превышает 10^{-12}.

Алгоритм Миллера–Рабина, как и многие другие рандомизированные алгоритмы проверки чисел на простоту, опирается на *малую теорему Ферма.*

Теорема 1. *Пусть p — простое число, а a — натуральное. Тогда*

$$(a^p - a) \bmod p = 0,$$

то есть $a^p - a$ делится на p.

Существует и другая, эквивалентная, формулировка малой теоремы Ферма.

Теорема 2. *Пусть p — простое число, а a — натуральное, причем a не делится на p. Тогда*

$$a^{p-1} \bmod p = 1.$$

Доказательство теоремы 1. Прежде всего мы докажем, что если $0 < k < p$, то C_p^k делится на p. Действительно,

$$C_p^k = \frac{p(p-1)\dots(p-k+1)}{k!}.$$

Поскольку числитель делится на p, а знаменатель — нет, то частное делится на p.

Теперь применим индукцию по a. Утверждение теоремі очевидно для $a = 1$. Предположим, что оно верно для некоторого $a = b$, докажем его для $a = b + 1$. Имеем

$$a^p - a = (b+1)^p - (b+1) = \sum_{k=0}^{p} C_p^k b^k - (b+1) = b^p - b + \sum_{k=1}^{p-1} C_p^k b^k.$$

Ясно, что $b^p - b$ делится на p по предположению индукции, а каждое слагаемое в последней сумме делится на p по доказанному выше. □

5. Сложные и простые алгоритмы

Сложность вычислительных алгоритмов обычно измеряется временем, необходимым для обработки входных данных. Если входными данными являются числа, то сложность алгоритма есть функция количества m десятичных цифр наибольшего из чисел.

Время выполнения *простых* алгоритмов яляется пропорциональной степени числа m или пропорциональной полиному от m. Например, перемножение двух m-значных чисел можно осуществить с помощью $O(m^2)$ алгоритма.

Алгоритм считается *сложным*, если время его выполнения растет быстрее, чем любая степень m. *Экспоненциальные* алгоритмы, для которых время выполнения растет как e^{am}, $a > 0$, считаются суперсложными и их выполнение считается крайне трудоемким.

5.1. Сложность алгоритма факторизации. Например, алгоритм факторизации числа n с помощью проверки делением на все числа, меньшие $\sqrt{n}$, является экспоненциальным: если число n состоит из m десятичных цифр, то необходимо осуществить $\sqrt{n} \approx 10^{m/2}$ или e^{am} проверок при $a = \frac{1}{2}\ln 10 \approx 1.15$.

Говорят, что алгоритм принадлежит классу **P**, если он выполняется за полиномиальное время (в таком случае мы также говорим, что алгоритм является полиномиальным). Все другие алгоритмы относятся к классу **NP**.

Прогресс в развитии математических методов имеет в рассматриваемых задачах намного более важное значение, чем увеличение быстродействия компьютеров. В 1977 году Рональд Ривест, один из авторов современной криптографии, писал, что для факторизации 125-значного числа с большими делителями необходимы 40 квадриллионов лет. Однако уже в 1994 году была осуществлена факторизация 129-значного числа.

Одним из признанных достижений компьютерных наук XXI столетия явилось доказательство, опубликованное в 2002 году тремя индийскими математиками Агарвалом, Каяном и Саксеном в статье под названием “*PRIMES is in* **P**”. Основной результат этой работы можно предсказать по ее названию: авторы предложили полиномиальный алгоритм проверки чисел на простоту.

6. Как проверить пароль, не зная его?

Описанную идею сохранения шахматного кода можно использовать и для более “серьезных” задач. Например, при выдаче денег из банкомата центральный компьютер банка может проверить пароль клиента, не зная самого пароля! На первый взгляд и эта задача кажется неразрешимой, но теория сложности поможет и тут.

В банковской практике встречаются случаи, когда клиенты хранят ценные документы в специальных ячейках. Клиенты не желают, чтобы банк (или кто-либо другой) знал пароль доступа к ячейке. В силу важности документов пароль меняется после каждого использования ячейки. Можно ли скрыть пароль от банка, но при этом иметь доступ к своей ячейке?

Предположим, что пароль — это 100-значное простое число p (такой пароль представить себе трудно; поэтому предположение не вполне реалистично, но позволяет проще рассказать о возможном решении). В тот момент, когда клиент выбирает пароль p, он выбирает также и другое простое число q, теперь 101-значное. Банку сообщается их произведение $N = pq$. В любой момент, когда клиент желает снять сумму со своего счета, он вводит пароль p, а компьютер банка проверяет является ли p делителем N. Проверка делимости одного числа на другое занимает считанные мгновения. Однако задачу нахождения делителей N можно решить только за экспоненциальное время, то есть практически никогда для 100-значных паролей. Это означает, что никто счетом клиента, кроме него самого, воспользоваться не может, даже зная число N, которое содержит всю информацию о пароле.

7. Еще одна экспоненциальная задача

Большое количество задач класса **NP** можно найти в теории графов. Среди них встречаются и такие задачи, что специалисты верят в их простоту, но в настоящее время не могут построить подходящий алгоритм. Одной из них, известной под именем задачи о сватовстве, можно придать форму занимательного рассказа.

7.1. Первая задача короля Артура. При дворе короля Артура было 150 неженатых рыцарей и 150 незамужних

дам. Королю пришло на ум переженить их, но дело осложнялось тем, что не все из них были согласны с предлагаемыми королем вариантами. Король призвал своего советника Мерлина и под страхом казни приказал ему найти вариант, который устроил бы все пары.

Мерлин сразу осознал невозможность простого перебора всех 150! вариантов для решения задачи. Однако, имея фантастические таланты, он в назначенный срок смог во всем блеске продемонстрировать их перед своим королем. Мерлин попросил 56 отобранных им дам встать с одной стороны королевского трона (критерий отбора знал только сам Мерлин); с другой стороны встали 95 отобранных Мерлином рыцарей. После этого королевский советник задал вопрос: “Желает ли кто-нибудь из 56 прекрасных дам выбрать себе в мужья одного из этих 95 достойных рыцарей?”. Когда дамы хором ответили “Нет!”, Мерлин обратился к Артуру: “Ваше величество, как можно найти мужей этим 56 дамам среди оставшихся 55 рыцарей?”.

Способ, которым Мерлин нашел комбинацию 56 дам и 95 рыцарей, остается загадкой и по сей день. Однако его рассуждения безупречны и абсолютно убедительны.

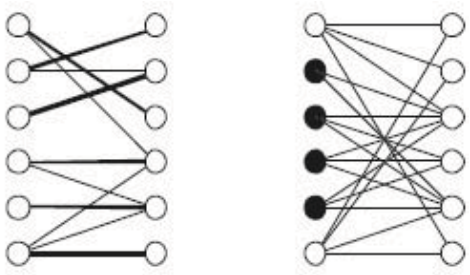


Рис. 2.
Два графа для задачи об
идеальном паросочетании

В этой задаче, как и в задаче о пароле для банка, про-

верка конкретного условия труда не представляет. Но нахождение самого этого условия является **NP** задачей!

Задача, которую мы только что рассмотрели, в теории графов называется задачей об *идеальном паросочетании*. Ниже представлены два примера графов, которые можно использовать в этой задаче. Для левого графа задача имеет решение, для правого — нет.

Каждый отрезок на представленных графах обозначает допустимое паросочетание. Для левого графа решение задачи изображено жирными отрезками. Для правого графа задача о паросочетании не имеет решения: 4 жирные точки соединены всего с тремя точками справа.

Описанная история при дворе короля Артура вскоре получила продолжение.

7.2. Вторая задача короля Артура. Через некоторое время король Артур стал замечать, что как ни рассади рыцарей за круглым обеденным столом, некоторые из соседей громко спорят и даже переходят на взаимные оскорбления. Впечатленный решением Мерлина в деле о сватовстве, король призвал его еще раз и приказал рассадить рыцарей так, чтобы раз и навсегда прекратить распри за столом. Мерлин моментально сообразил, что ни один из имеющихся 150! способов не приведет к успеху. Королю же он сказал следующее: “Ваше величество, Ваше желание прекратить надоедливые споры похвально в высшей степени: что может быть важнее мира и покоя при приеме пищи? Однако выполнить его невозможно ни при каких обстоятельствах. Если бы был такой рыцарь, у которого только один друг, невозможность решения была бы очевидной, поскольку каждый имеет двух соседей. Однако каждый из этих достойных рыцарей, Ваше величество, имеет много больше друзей и тем не менее миролюбивое расположение все же невозможно. Я могу доказать это, но мои объяснения

будут слишком сложны и займут слишком много Вашего драгоценного времени. Они могут даже продолжаться и до конца жизни”. Не впечатленный такой перспективой, король Артур, поколебавшись немного, все же отказался от своей миротворческой затеи.

В теории графов вторую задачу короля Артура называют нахождением *гамильтонового пути*. Многие считают эту задачу полиномиальной, но алгоритм ее решения за полиномиальное время пока не известен.

УПРАЖНЕНИЯ

Упражнение 1. Пусть p — это 200 значное, а q — 201 значное простые числа. Доказать, что pq — это число, имеющее 400 или 401 десятичных цифр.

Упражнение 2. Написать программу для компьютера, которая генерирует m случайных 200-значных чисел (m — это параметр программы). Запуская программу для разных m, оценить время, необходимое `Алисе`, чтобы получить случайное m-значное простое число.

Упражнение 3. Написать программу для компьютера, которая реализует алгоритм 1. Запуская программу для разных n, оценить надежность, с которой алгоритм Рабина–Миллера определяет простоту числа n.

Упражнение 4. Доказать, что теоремы 1 и 2 эквивалентны.

Упражнение 5. Привести пример составного p, для которого малая теорема Ферма (теорема 1) не верна.

Упражнение 6. Пусть натуральное число a не делится на простое p. Обозначим через r_i остаток от деления ia на p. Доказать, что $r_1 r_2 \dots r_{p-1} = (p-1)!$.

Упражнение 7. (продолжение) Пусть натуральное число a не делится на простое p. Обозначим через r_i остаток от деления ia на p. Доказать, что $a \cdot (2a) \cdot (3a) \cdot \ldots \cdot ((p-1)a) = r_1 r_2 \dots r_{p-1} \pmod p = (p-1)! \pmod p$.

Упражнение 8. (продолжение) Пусть натуральное число a не делится на простое p. Используя упражнения 6 и 7, дать другое доказательство малой теоремы Ферма (теоремы 1).

Упражнение 9. Пусть a — натуральное число, а p — простое. Рассмотрим все числа, которые можно записать в системе по основанию a с помощью не более, чем p символов ("*цифр*"). Те из них, запись которых требует менее, чем p цифр, дополним слева нулями. Для каждого такого $x = c_{p-1}c_{p-2}c_{p-3}\ldots c_2c_1c_0$ рассмотрим совокупность тех чисел, которые получаются из него циклическим сдвигом его цифр, например $y = c_{p-2}c_{p-3}\ldots c_2c_1c_0c_{p-1}$ или $z = c_{p-3}\ldots c_2c_1c_0c_{p-1}c_{p-2}$. Два числа, получаемые друг из друга циклическим сдвигом цифр, отнесем к одному классу. Какие при этом два типа классов возникают? Сколько элементов принадлежат каждому из классов?

Упражнение 10. (продолжение) Доказать, что если не все цифры числа $x = c_{p-1}c_{p-2}c_{p-3}\ldots c_2c_1c_0$ одинаковые, то все циклические сдвиги цифр приводят к разным числам.

Упражнение 11. (продолжение) Используя упражнения 9 и 10, доказать малую теорему Ферма (теорему 1).

Упражнение 12. Доказать, что перемножение двух m-значных чисел можно осуществить с помощью $O(m^2)$ алгоритма.

Упражнение 13. Пусть $p > 2$ — простое и $1 \le a \le p-1$. Тогда уравнение $ax = 1 \pmod p$ имеет единственное решение $a' \in \{1, 2, \ldots, p-1\}$.

Упражнение 14. (продолжение) Доказать, что если $a = a'$, то $a \in \{1, p-1\}$.

Упражнение 15. (продолжение) Доказать, что $2 \cdot 3 \cdot \ldots \cdot (p-2) = 1 \pmod p$.

Упражнение 16. (продолжение) Используя упражнения 13–15, доказать теорему Вильсона: *если p простое число, то*

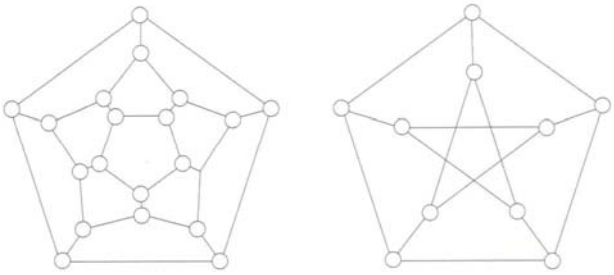
$$(p-1)! = -1 \pmod p. \tag{1}$$

Упражнение 17. Предположим, что для некоторого p выполнено (1). Если $p \ge 4$ составное, то выберем один из его делителей $1 < q < p$. Доказать, что $(p-1)!$ делится на q.

Упражнение 18. (продолжение) Используя упражнение 17, доказать, что если (1) выполнено для некоторого p, то p — простое число.

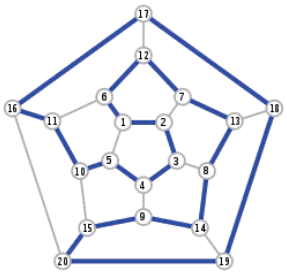
Упражнение 19. Доказать, что существует 150! вариантов в задаче о паросочетании для 150 пар.

Упражнение 20. *Гамильтоновым* называется такой путь в графе, который включает каждую вершину только один раз. Задача нахождения гамильтонового пути в графе является сложной. Найти гамильтонов путь (или доказать, что его нет) в следующих графах



Эти знаменитые графы имеют собственные имена: граф слева называется *додекаэдром*, а справа — *графом Петерсена*.

Ответ. Гамильтонов путь (даже цикл) для додекаэдра показан ниже.



В графе Петерсена нет гамильтонового пути.

8. Заключительные замечания

Материал лекции основан на части главы 15 из книги

L. Lovász, J. Pelikán, and K. Vesztergombi, "*Discrete Mathematics*", Springer-Verlag, Berlin, 2003.

1. Алиса и Боб традиционные персонажи, которые оживляют изложение материала по шифрованию или криптографии. Вполне можно было бы заменить их на A и B. Автором рис. 1 является Вера Изюмцева (14.12.2009 г.).

2. Самое эмоциональное описание функции $\pi(x)$ дано Дон Цагиром: "Простые числа ... являются самыми капризными и упрямыми из всех объектов, вообще изучаемых математиками. Они растут среди натуральных чисел как сорная трава, не подчиняясь, кажется, ничему, кроме случая, и никто не может предсказать, где взойдет еще одно простое число, а увидев число — определить простое оно или нет. Другой факт озадачивает еще больше, так как он состоит в прямо противоположном утверждении, а именно: простые числа демонстрируют удивительную регулярность, они подчиняются законам, и притом с почти педантической точностью."[2]

3. Во введении к своему трактату *De numeris amicabilibus* ("*О дружественных числах*") Леонард Эйлер писал: "Из всех проблем, рассматриваемых в математике, нет таких, которые считались бы в настоящее время более бесплодными и бесполезными, чем проблемы, касающиеся *природы чисел* и их *делителей*. В этом отношении нынешние математики сильно отличаются от древних, придававших гораздо большее значение исследованиям такого рода ... А именно, они не только считали, что отыскание истины похвально само по себе и достойно человеческого познания, но, кроме того, совершенно справедливо полагали, что при этом замечательным образом развивается изобретательность и перед человеческим разумом раскрываются новые возможности решать сложные задачи ... Математика, вероятно, никогда не достигла бы такой высокой степени совершенства, если бы древние не приложили столько усилий для изучения вопросов, которыми сегодня пренебрегают из-за их мнимой бесплодности."

4. Невозможно устоять перед искушением процитировать по этому поводу несколько фраз из "*Disquisitiones Arithmeticae*" ("*Арифметические исследования*") К. Гаусса: "Что задача различать простые и

[2] Цитата из сборника "*Живые числа*", изд-во "Мир", Москва, 1985.

составные числа, а последние разлагать на простые множители, принадлежит к важнейшим и полезнейшим задачам всей арифметики и что она занимала ум как древних, так и современных математиков, настолько известно, что было бы излишним тратить на это много слов. Тем не менее следует признать, что все до сих пор предложенные методы или ограничиваются частными случаями, или настолько громоздки и трудоемки, что ... в основном едва ли могут быть применимы ... к большим числам ... ; достоинство науки требует, чтобы прилежно усовершенствовались все вспомогательные средства, могущие помочь в решении этой знаменитой проблемы.”

5. За свое достижение Агарвал, Каян и Саксена получили в 2006 году премии Геделя и Фулкерсона в области компьютерных наук.

6. Пароли широко используются компьютерами и программами, например архиваторами. Среди трех самых популярных архиваторов, ARJ, ZIP и RAR, последний безусловно обеспечивает самое стойкое шифрование, потому что он использует стойкий алгоритм AES и скорость перебора его паролей самая низкая. К сожалению, это не значит, что при реализации шифрования в нем не допущены какие-либо ошибки, так как исходные тексты RAR закрыты и они не были проанализированы криптоаналитиками. Среди архиваторов, использующие стойкие алгоритмы и распространяющихся с исходными текстами, можно выделить 7-Zip, но также неизвестно, был ли проведен их анализ с криптографической точки зрения.

7. Задачи короля Артура описаны в книге

> Л. Ловаса и М. Пламмера, “*Прикладные задачи теории графов. Теория паросочетаний в математике, физике, химии*”, изд-во ‘‘Мир’’, Москва, 1998.