

1 Необратимые и односторонние функции

1.1 Определения

Пусть имеется семейство функций $f_n: \mathbb{B}^{k(n)} \rightarrow \mathbb{B}^{l(n)}$, где k, l некоторые полиномы.

Мы хотим определить понятие “плохо обратимой” функции. Обратить $f(x)$ означает найти какое-то x' такое, что $f(x') = f(x)$. Возможны разные варианты определения: слабый и сильный. Неформально говоря, обращение слабо необратимой функции оказывается неверным с заметной (не пренебрежимо малой) вероятностью; для сильно необратимой функции оно неверно почти всегда (вероятность удачного обращения пренебрежимо мала). В качестве процедур обращения мы будем рассматривать последовательности схем C_n из функциональных элементов, размер которых ограничен некоторым полиномом от n . Схема C_n должна иметь $l(n)$ входов и $k(n)$ выходов.

Дадим точные определения.

Семейство функций f_n описанного вида называется *слабо необратимым*, если существует некоторый многочлен p с таким свойством: для любой последовательности схем C_n из функциональных элементов, размер которых ограничен некоторым полиномом от n вероятность события

$$f_n(C_n(f_n(x))) = f_n(x),$$

(где все слова x длины $k(n)$ считаются равновероятными) не превосходит

$$1 - \frac{1}{p(n)}$$

для всех достаточно больших n . Если $f_n(C_n(f_n(x))) = f_n(x)$, то мы говорим, что схема C_n успешно обращает $f_n(x)$.

В этом определении на вход схемы C_n (гипотетического алгоритма обращения) дается слово длины $l(n)$, и она должна найти слово длины $k(n)$, попадающее в f_n -прообраз входного слова. Такой прообраз существует, так как обращению подлежит слово вида $f_n(x)$ для некоторого (неизвестного алгоритму) слова x .

Важно, что мы не требуем точного обращения (равенства $C_n(f_n(x)) = x$). Если бы требовали, то любая функция, склеивающая все слова в одно,

оказалась бы необратимой.) Ещё отметим, что равномерное распределение берётся на словах x , а не на их образах (иначе трудность обращения была бы связана с тем, что многие слова не имеют прообраза).

Наконец, заметим, что многочлен p , задающий долю неудачных попыток обращения, не должен зависеть от последовательности схем C_n . Если оказывается, что с ростом сложности схем (в пределах полиномиальной) обращение становится возможным всё лучше и лучше, и ошибка может быть сделана меньше любого наперёд заданного полинома, то это не годится.

Семейство функций f_n называется *сильно необратимым*, если для любой последовательности схем C_n из функциональных элементов, размер которых ограничен некоторым полиномом от n , вероятность события

$$f_n(C_n(n, f_n(x))) = f_n(x),$$

(где все слова x длины $k(n)$ считаются равновероятными) стремится к нулю при $n \rightarrow \infty$ быстрее любого обратного полинома. Это означает, что для любого многочлена q она не превосходит

$$\frac{1}{q(n)}$$

для всех достаточно больших n . (Разумеется, та граница, с которой начинаются “достаточно большие” n , может зависеть от q .)

Семейство f_n называется полиномиально вычислимым, если имеется алгоритм, который по n вычисляет $k(n)$ и $l(n)$, а также имеется полиномиальный алгоритм, получающий на вход n и слово x длины $k(n)$ и вычисляющий (за полиномиальное от n время) слово $f_n(x)$. Необратимые полиномиально вычисляемые семейства функций называются односторонними. Аналогично, слабо необратимые семейства, вычисляемые за полиномиальное время называются слабо односторонними.

В дальнейшем мы будем позволять себе следующую терминологическую вольность: мы будем говорить “[слабо] необратимая функция” вместо “[слабо] необратимое семейство функций” и “[слабо] односторонняя функция” вместо “[слабо] одностороннее семейство функций”.

Задача 1. (1) Докажите, что определение слабо необратимой функции можно переформулировать следующим образом: для некоторого полинома p для любого полинома q лишь для конечного числа n существует схема размера $q(n)$, успешно обращающая $f_n(x)$ с вероятностью не менее

$1 - 1/p(n)$. (2) Докажите, что определение сильно необратимой функции можно переформулировать следующим образом: для любого полинома q существует последовательность чисел ε_n , стремящаяся к нулю быстрее любого обратного полинома от n и такая, что лишь для конечного числа n существует схема размера не более $q(n)$, успешно обращающая $f_n(x)$ с вероятностью более ε_n .

Задача 2. Пусть для бесконечно многих n для всех слов x длины $k(n)$ выполнено $f_n(x) = 00 \dots 0$. Докажите, что тогда f_n не является слабо обратимой. Докажите, что функция не является слабо необратимой даже, если мощность множества значений f_n ограничена некоторым полиномом от n (для бесконечно многих n).

Задача 3. Пусть даны сильно [слабо] необратимая функция $f_n : \{0, 1\}^{k(n)} \rightarrow \{0, 1\}^{l(n)}$ и числовая функция $m(n)$, ограниченная многочленом и вычисляемая за полиномиальное от n время. Докажите, что тогда функция $xu \mapsto f_n(x)u$ определенная на словах длины $k(n) + m(n)$ сильно [слабо] необратима.

Задача 4. Докажите, что необратимые функции существуют.

Задача 5. Пусть даны две биективные полиномиально вычисляемые функции $f_n, g_n : \{0, 1\}^{k(n)} \rightarrow \{0, 1\}^{k(n)}$. Докажите, что если хотя бы одна из них сильно [слабо] необратима, то и их композиция сильно [слабо] необратима.

Задача 6. Докажите, что существуют сильно необратимые функции $f_n, g_n : \{0, 1\}^{k(n)} \rightarrow \{0, 1\}^{k(n)}$, композиция которых не является даже слабо необратимой (полиномиальная вычислимость функций не требуется).

Задача 7. Докажите, что существуют слабо необратимые функции, не являющиеся сильно необратимыми.

Задача 8. Пусть $f_n : \{0, 1\}^n \rightarrow \{0, 1\}^n$ выбирается случайно среди всех функций (необязательно биективных). Все функции считаются равновероятными и f_n независимы при разных n . Докажите, что с вероятностью 1 семейство f_n необратимо.

Задача 9. Докажите, что если функция f_n необратима, то и функция $g_n(x) = f_n(x)0$ необратима.

Задача 10. Докажите, что существует необратимая функция f_n такая, что функция $x \mapsto (f_n(x)$ без последнего бита) обратима.

Задача 11. Докажите, что существуют функции f_n, g_n , не являющиеся слабо необратимыми и такие, что функция $x \mapsto f_n(x)g_n(x)$ сильно необратима.

Задача 12. Докажите, что если функция f_n сильно [слабо] необратима, то и функция $x \mapsto f_n(x)f_n(x)$ сильно [слабо] необратима.

Задача 13. Докажите, что существуют сильно необратимые функции f_n, g_n такие, что функция $x \mapsto f_n(x)g_n(x)$ не является даже слабо необратимой.

Назовем вероятностной схемой любую схему C , входы которой разделены на две группы, y_1, \dots, y_k и r_1, \dots, r_s , называемых основными и случайными входами.

Задача 14. Докажите, что если функция f_n [слабо] необратима, то для любой последовательности вероятностных схем C_n , размер которых ограничен полиномом от n , с $l(n)$ основными входами и $k(n)$ выходами, вероятность события $f_n(C_n(f_n(x))) = f_n(x)$ стремится к нулю быстрее любого обратного полинома от n [не больше $1 - 1/s(n)$ для некоторого многочлена r для всех достаточно больших n]. Здесь мы считаем, что все слова x длины $k(n)$ считаются равновероятными, а на случайные входы схемы подаются независимые от x и друг от друга исходы бросаний симметричной монетки.

Неизвестно, существуют ли односторонние или хотя бы слабо односторонние функции. Доказать их существование сложно, поскольку из него следует, что $P=NP$.

Теорема 1. *Если $P=NP$, то любое полиномиально вычислимое семейство функций f_n , не является слабо необратимым. Более того, существует алгоритм, который для всех x по n и $f_n(x)$ за полиномиальное от n время находит некоторый прообраз $f(x)$ длины $k(n)$.*

Доказательство. Рассмотрим следующий язык L из NP . Он состоит из всех троек слов $(1^n, y, z)$, таких, что длина z равна $l(n)$ и найдется слово x длины $k(n)$, начинающееся на z , для которого $f_n(x) = y$. Поскольку мы предположили, что $P=NP$, этот язык можно разрешить за полиномиальное от n количество шагов. Пусть нам даны n и слово y длины $l(n)$. Сначала выясняем, принадлежит ли L тройка $(1^n, y, \text{пустое слово})$. Если нет, то y не имеет прообраза. Иначе запускаем бинарный поиск прообраза y . А именно, сначала выясняем принадлежит ли L тройка $(1^n, y, 0)$. Если

да, то у y есть прообраз, начинающийся с нуля, а иначе — с единицы. И так далее. После i шагов у нас имеется строка z длины i , для которой мы знаем, что y имеет прообраз, начинающийся на z . После $k(n)$ шагов мы найдем некоторый прообраз целиком. \square

Приведем три семейства функций, которые возможно являются односторонними.

Произведение натуральных чисел: $f_n(x)$ равно произведению первой и второй половинок x , рассматриваемых как двоичные записи натуральных чисел. Здесь $k(n) = l(n) = 2n$ (произведение двух n -битовых чисел содержит не более $2n$ битов).

Функция SUBSET-SUM. Определение этой функции навеяно NP-полной задачей о сумме подмножеств (SUBSET-SUM). Здесь $k(n) = n^2 + n$, $l(n) = n^2 + 2n + \lceil \log n \rceil$. Значение f_n на словах x длины $n^2 + n$ определяется так. Разрежем x на $n + 1$ блоков длины n и будем понимать первые n блоков как n натуральных чисел x_1, \dots, x_n в двоичной записи. Последний блок будем понимать как подмножество I множества $\{1, 2, \dots, n\}$. По определению $f_n(x)$ равно конкатенации x_1, \dots, x_n и $\sum_{i \in I} x_i$.

Экспонента в кольце вычетов (функция Блюма и Микэли). Функция f_n определена на словах длины $3n$. Ее значение на слове xuz , где x, y, z слова длины n , понимаемые как двоичные записи натуральных чисел, равно конкатенации $x^z \bmod y$, x и z . То есть, обращение является логарифмированием, а не извлечением корня. Задача обращения функции Блюма и Микэли называется дискретным логарифмированием.

Задача 15. Пусть $f_n : \{0, 1\}^{k(n)} \rightarrow \{0, 1\}^{l(n)}$ одностороннее семейство такое, что $k(n)$ строго возрастает. Докажите, что существует одностороннее семейство $g_m : \{0, 1\}^m \rightarrow \{0, 1\}^*$ такое, что $f_n = g_{k(n)}$ для всех n . (Семейство g_m можно было бы назвать всюду определенным односторонним продолжением f_n .)

Как говорилось, существование сильно односторонних семейств является лишь гипотезой. (Эта гипотеза лежит в основе почти всех криптографических протоколов.) Но оказывается, что из слабо одностороннего семейства можно получить сильно одностороннее.

1.2 Усиление

Теорема 2. *Если существуют слабо односторонние функции, то существуют и сильно односторонние функции.*

(Заметим, что в этом случае существуют и слабо односторонние функции, не являющиеся сильно односторонними, см. задачу 7.)

Доказательство. Пусть f_n — слабо одностороннее семейство и $p(n)$ такой полином, что вероятность успешного обращения $f_n(x)$ схемами размера $\text{poly}(n)$ при достаточно больших n не превосходит $1 - 1/p(n)$. Для краткости мы будем опускать индекс n и говорить об одной функции. (Имеется в виду, что это построение и все рассуждения выполняются параллельно для всех n .) Рассмотрим новую функцию f^N , для которой

$$f^N(x_1x_2\dots x_N) = f(x_1)f(x_2)\dots f(x_N).$$

Она определена на словах длины $Nk(n)$, которые отождествляются с кортежами из N слов длины $k(n)$, и принимает значения длины $Nl(n)$. В качестве N мы возьмем некоторый достаточно большой многочлен от n (какой — скажем дальше).

Начнём с простого (но, увы, неправильного) объяснения, почему при достаточно большом N функция f^N является сильно односторонней. В самом деле, обращение $f^N(x_1\dots x_N)$ для случайного $x_1\dots x_N$ требует одновременного обращения $f(x_1), \dots, f(x_N)$ при независимых случайных x_1, \dots, x_N . В каждом из N случаев вероятность ошибки не меньше $1/p(n)$. Поэтому общая вероятность успеха не больше

$$\left(1 - \frac{1}{p(n)}\right)^N.$$

Если положить $N = np(n)$, то выражение это равно $(1 - 1/p(n))^{p(n)}$ в степени n , что экспоненциально убывает (примерно равно e^{-n}). Поэтому функция f^N сильно односторонняя. Забегая вперед, отметим, что хотя это рассуждение и ошибочное, выбранное значение N окажется правильным.

Исправление

Что неверно в этом рассуждении? Дело в том, что мы рассматриваем вероятность успешного обращения f^N с помощью алгоритма специального вида, состоящего в том, что некий алгоритм обращения функции f применяется параллельно для всех n . А что делать, если алгоритм обращения такого вида не имеет? Кажется странным, что какие-то алгоритмы обращения функции f^N могут делать что-то более сложное,

чем обращаться в отдельности каждое $f(x_i)$, но это не доказательство (и непонятно, как это можно было бы формализовать буквально).

Правильное доказательство должно идти в другом направлении: вместо того, чтобы начинать с алгоритма обращения для f и затем смотреть, что получится при его раздельном применении к $f(x_i)$, мы должны рассмотреть произвольный алгоритм R , претендующий на обращение функции f^N , и показать, что если он имеет непренебрежимые шансы на успех, то для функции f есть алгоритм R' , у которого вероятность ошибки обращения меньше $1/p(n)$. Заметим, что алгоритм R' и реализующая его схема из функциональных элементов по задаче 14 могут быть вероятностными.

Попробуем следующий (ещё не окончательный) вероятностный алгоритм обращения $f(x)$. Пусть нам дано некоторое слово y , прообраз которого мы ищем. Мысленно представим себе, что y есть $f(x_1)$ для неизвестного x_1 и дополним слово y словами $f(x_2), \dots, f(x_N)$ для случайных слов x_2, \dots, x_N . Применим к полученному слову

$$yf(x_2) \dots f(x_N)$$

алгоритм R и посмотрим, что он даст на первом месте (каковы первые $k(n)$ битов). Это слово и будет результатом работы алгоритма.

Нетрудно понять, что вероятность успеха алгоритма R' (для $f(x_1)$ при случайном x_1) не меньше вероятности успешного обращения $f(x_1) \dots f(x_N)$ алгоритмом R для случайных $x_1 \dots x_N$. Этого нам мало, поскольку надо перейти от не очень малой вероятности к вероятности, близкой к единице.

Как можно усовершенствовать алгоритм R' ? Сразу возникают две идеи. Во-первых, можно производить несколько попыток обращения подряд (и выбирать из них удачную, если таковая случилась). Ведь сама функция f полиномиально вычислима, поэтому мы можем за полиномиальное время проверить, удалась ли попытка обращения. Другая идея: мы можем ставить обрабатываемое слово y не только на первое место, но и на любое другое место от 1 до N .

Алгоритм обращения

Этих двух идей окажется достаточно для доказательства теоремы. А именно, для данной схемы R , претендующей на обращение f^N , мы рас-

смотрим вероятностную схему для обращения $f(x)$, действующую следующим образом.

Для каждого i от 1 до N поставим обращаемое слово y на i -е место и окружим его $N - 1$ словами, получив набор

$$f(x_1), \dots, f(x_{i-1}), y, f(x_{i+1}), \dots, f(x_N).$$

Слова $x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_N$ выбираются независимо и равномерно распределены в $\mathbb{B}^{k(n)}$. К полученному набору применяется схема R . Если i -я компонента результата лежит в прообразе y , констатируем удачу (и сворачиваем всё дальнейшее).

Эти действия (N попыток, по одной для каждого i) составляют один этап алгоритма R' . Полностью R' состоит в повторении этого действия несколько раз. Обозначим число повторений через M (и выберем его дальше).

[Заметим сразу, что ставить y на разные позиции необходимо. В самом деле, если R почему-то не работает на образах слов, в которых первый бит (первого слова) равен нулю, то он вполне может иметь вероятность успеха $1/2$. Но тогда и алгоритм R' (если y ставится лишь на первое место) не будет работать на образах таких слов, и вероятность успеха не подымется выше $1/2$, сколько повторений ни делай.]

Теперь нам нужно подобрать значение M . (Значение M может зависеть только от n , функции f_n и схемы R .) Нам нужно, чтобы из того, что вероятность успеха R' меньше $1 - 1/p(n)$, следовало бы, что вероятность успеха R меньше $1/q(n)$. Для того чтобы подобрать M с этим свойством, необходимо разобраться, как связаны вероятности успеха для алгоритмов R и R' . Обозначим через $s_1(x_1)$ вероятность того, алгоритм R правильно обращает слово

$$f(x_1)f(x_2)\dots f(x_N)$$

для случайно выбранных x_2, \dots, x_N . Вероятность успешного обращения для первой попытки (когда мы ставим y на первое место) для слова $f(x_1)$ не меньше $s_1(x_1)$. Аналогичные вероятности можно рассмотреть и для других попыток (всего в каждой фазе N попыток). По аналогии обозначим их $s_2(x_2), \dots, s_N(x_N)$.

Какова вероятность успеха на одной фазе? Для входа $f(x)$ она заведомо не меньше

$$\max(s_1(x), \dots, s_N(x)),$$

поскольку успех одной попытки означает успех всей фазы. (Более точно сказать трудно, эта вероятность, вообще говоря, не определяется значениями $s_i(x)$, поскольку успехи в попытках зависимы: может оказаться, скажем, что некоторые слова трудны для обращения во всех координатах.) Обозначим этот максимум через $\hat{s}(x)$.

Дальнейший ход доказательства можно описать так. Посмотрим, сколько имеется слов x , для которых величина $\hat{s}(x)$ очень мала (меньше некоторой границы ε , которую мы выберем впоследствии). Такие слова, неформально говоря, трудны для обращения. Нетрудные слова хорошо обращаются алгоритмом R' , поскольку на каждой фазе вероятность обращения не меньше ε , фазы независимы и их много. Пользуясь тем, что вероятность неудачи R' больше $1/p(n)$ мы докажем, что трудных слов много. Отсюда будет следовать R обращает f^N с вероятностью меньше $1/q(n)$. Эти неформальные разговоры следует, конечно, уточнить.

Оценки

Пусть ε — некоторое положительное число, порядка $1/\text{poly}(n)$. Будем называть слово x “трудным”, если $\hat{p}(x) < \varepsilon$. Обозначим через δ долю трудных слов среди всех слов длины n . Тогда вероятность ошибки при обращении слова $f(x)$ (для случайного x) меньше

$$\delta + (1 - \varepsilon)^M.$$

Положим $M = n/\varepsilon$, второе слагаемое тогда примерно равно e^{-n} .

Алгоритм R' реализуется вероятностной схемой, которая по существу состоит из MN схем R и такого же количества схем для вычисления f_n , а значит реализуется схемой размера, ограниченного многочленом от n . По условию вероятность ошибки R' не меньше $1/p(n)$ (если n достаточно велико). Поэтому при всех достаточно n выполнено $\delta \geq 1/2p(n)$.

Выделим из трудных слов часть, образующую долю ровно $1/2p(n)$. Оценим сверху вероятность успешного обращения $f^N(x_1, \dots, x_N)$ алгоритмом R на случайном входе. Оценивая её, разделим пространство на две части: когда одно из слов x_i является выделенным и когда все они не выделены. Для второй части нам не важны значения $\hat{s}(x)$, мы пользуемся тем, что эта часть сама по себе имеет вероятность не больше $(1 - 1/2p(n))^N$ (каждое из x_i выделено с вероятностью $1/2p(n)$, а всего их N и они независимы).

Первая часть: мы оценим вероятность в случае, когда выделенным (а значит трудным) оказалось слово x_i , а затем умножим оценку на N (вероятность объединения событий не больше суммы вероятностей). Вероятность того, что слово x_i окажется выделенным, равна $1/2p(n)$, а для каждого из выделенных слов x_i условная вероятность успешной работы алгоритма R (при данном значении i -й координаты) меньше ε . В итоге получаем для первой части оценку $N\varepsilon/2p(n)$, а для суммы (вероятности успешной работы R на случайном входе) оценку

$$N\varepsilon/2p(n) + (1 - 1/2p(n))^N.$$

Нам надо выбрать ε, N так, чтобы эта величина была строго меньше $1/q(n)$. Как было уже обещано, положим $N = np(n)$. Тогда второе слагаемое в этой сумме равно

$$\left(1 - \frac{1}{2p(n)}\right)^{np(n)} = \left[\left(1 - \frac{1}{2p(n)}\right)^{2p(n)}\right]^{n/2} \approx e^{-n/2}$$

а значит при больших n намного меньше $1/q(n)$. Поэтому ε можно определить так, чтобы первое слагаемое было равным $1/2q(n)$, то есть положить

$$\varepsilon = \frac{1}{nq(n)}.$$

Теорема доказана.

1.3 Необратимые частичные функции

Мы рассматривали односторонние функции f_n , определенные на множестве слов данной длины $k(n)$. В этом разделе мы рассмотрим функции определенные на некотором подмножестве D_n множества $\{0, 1\}^{k(n)}$. Такие функции будем называть *частичными*. Необратимость частичных функций определяется аналогично необратимости всюду определенных функций, только теперь обращающей схеме на вход подается $f_n(x)$ для случайным образом выбранного $x \in D_n$ и все слова из D_n считаются равновероятными. Односторонней частичной функцией будем называть полиномиально вычислимую необратимую функцию, для которой можно за полиномиальное от n время породить элементы из D_n так, чтобы все элементы порождались с примерно одинаковой вероятностью. Уточним последнее.

Последовательность распределений вероятностей μ_n на множестве двоичных слов называется *генерируемой за полиномиальное время*, если существует полиномиальный вероятностный алгоритм K такой, что для всех $x \in \{0, 1\}^*$ выполнено

$$\Pr[K \text{ выдает } x \text{ на входе } n] = \mu_n(x).$$

Алгоритм K получает на вход число n в унарной записи и при любых исходах своих бросаний должен остановиться за полиномиальное от n время и выдать некоторую двоичную строку.

Статистическим расстоянием между распределениями вероятностей μ и ν на множестве двоичных слов называется максимум $|\mu(A) - \nu(A)|$ по всем множествам слов A . Нетрудно понять, что этот максимум достигается на множестве слов $A = \{x \mid \mu(x) > \nu(x)\}$ (а также на его дополнении) и равен $\sum_{x \in \{0,1\}^*} |\mu(x) - \nu(x)|/2$. Последовательности распределений вероятностей μ_n и ν_n , называются *статистически неотличимыми*, если статистическое расстояние между μ_n и ν_n стремится к нулю быстрее любого обратного многочлена от n . Аналогично определяется статистическая неотличимость случайных величин: случайные величины α_n и β_n , зависящие от натурального параметра n , называются *статистически неотличимыми*, если их распределения $\mu_n(x) = \Pr[\alpha_n = x]$ и $\nu_n(x) = \Pr[\beta_n = x]$ статистически неотличимы.

Будем называть распределение μ_n *доступным*, если μ_n статистически неотлично от некоторого полиномиально генерируемого распределения ν_n .

Определение 1. Семейство частичных функций f_n называется *сильно односторонним*, если f_n полиномиально вычислимо, сильно необратимо и равномерное распределение на μ_n на области определения f_n доступно. Аналогично определяются слабо необратимой семейства частичных функций.

Задача 16. Докажите, что существует семейство функций g_n , не являющаяся сильно необратимым и такое, что каждую функцию можно сузить на некоторое множество, получив сильно необратимое семейство.

Задача 17. Предположим, существует сильно односторонняя частичная функция. Докажите, что тогда существует и сильно односторонняя всюду определенная функция.

Задача 18. Предположим, существует слабо односторонняя частичная функция. Докажите, что тогда существует и слабо односторонняя всюду определенная функция.

Задача 19. Предположим, существует слабо односторонняя частичная функция. Докажите, что тогда существует и сильно односторонняя всюду определенная функция.

Задача 20. Докажите, что существует всюду определенная функция не являющаяся слабо необратимой, сужение которой на некоторую последовательность подмножеств является сильно необратимым.

Задача 21. Докажите, что существует слабо необратимая частичная функция, сужение которой на любую последовательность множеств не является сильно необратимым.

Примеры предположительно сильно односторонних частичных функций

Функция Рабина. Функция Рабина f_n определена на словах xy (имеется в виду конкатенация слов) длины $4n$, где $|x| = |y| = 2n$, причем y есть двоичная запись числа $p \cdot q$, где p, q простые n -битовые числа вида $4k + 3$, а x произвольное $2n$ -битовое число. Ее значение равно конкатенации $x^2 \bmod y$ и y (слова x, y понимаются как двоичные записи натуральных чисел). Алгоритм генерации случайной величины, статистически неотличимой от случайной величины, равномерно распределенной в области определения f_n , работает так. Выбираем случайно число из n битов и проверяем, просто ли оно, с помощью полиномиального алгоритма проверки простоты. Если оно просто, положим p равным этому числу. Иначе повторяем попытку и т.д. Если после n^2 попыток простое число не найдется, то положим $p = 0$. Из закона распределения простых чисел следует, что вероятность этого события будет меньше $(1 - \varepsilon/n)^{n^2} < e^{-\varepsilon n}$ для некоторого положительного ε . Затем так же генерируем q . Число x выбираем равномерно среди всех $2n$ -битовых чисел. (Можно вместо алгоритма трех индийцев использовать и вероятностный тест на простоту с вероятностью ошибки, скажем 2^{-n} . Тогда статистическое расстояние будет больше на величину, равную вероятности того, что p или q составные, то есть 2^{-n+1} .)

Функция RSA. Функция RSA есть обобщение функции Рабина. Она определена на словах вида xuz , где x, y и z имеют длину $2n$ и понима-

ются как двоичные записи натуральных чисел, причем $y = p \cdot q$, p, q — простые n -битовые числа. Ее значение равно конкатенации $(x^z \bmod y)$, y и z . Доступность равномерное распределения доказывается так же, как и для функции Рабина. Отличие от функции Блюма и Микэли в том, что при обращении нужно найти основание степени, а не ее показатель.

Еще одно обобщение: неравномерные распределения на входах

В задаче обращения функции f_n мы до сих пор рассматривали только равномерные распределения на области определения f_n или статистически неотличимые от них распределения. Можно рассмотреть и произвольные доступные распределения. Будем называть случайную величину α_n *трудной* для f_n , если для любой последовательности схем C_n полиномиального от n размера вероятность события “ $C_n(\alpha_n) \in f_n^{-1}(f_n(\alpha_n))$ ” стремится к нулю быстрее любого обратного полинома от n . В этом определении мы не требуем, чтобы α_n всегда принимала значения в области определения f_n ; если $f_n(\alpha_n)$ не определено, то мы считаем, что указанное событие выполнено, поэтому вероятность этого должна быть ничтожно мала. Если частичная функция имеет доступную трудную случайную величину, то будем называть ее *обобщенно необратимой*. Это понятие обобщает понятие сильно необратимой функции. Аналогичным образом можно обобщить понятие слабо необратимой функции, но это обобщение нам не понадобится в дальнейшем.

Задача 22. Докажите, что если существует обобщенно односторонняя функция (обобщенно необратимая полиномиально вычислимая частичная функция), то существует и сильно односторонняя всюду определенная функция.

2 Генераторы псевдослучайных чисел

2.1 Вычислительно неотличимые случайные величины

Случайные величины α_n и β_n , зависящие от натурального параметра n , со значениями в множестве слов некоторой длины $l(n)$ называются вычислительно неотличимыми, если для любой последовательности схем

C_n размера $\text{poly}(n)$ (с $l(n)$ входами и одним выходом) вероятность событий $C_n(\alpha_n) = 1$ и $C_n(\beta_n) = 1$ отличаются на пренебрежимо малую величину. Схема C_n в этом контексте называется тестом и мы говорим, что случайная величина α_n проходит тест C_n , если $C_n(\alpha_n) = 1$. Таким образом, мы требуем, чтобы α_n и β_n проходили любые тесты полиномиального размера с приблизительно равной вероятностью.

Если α_n и β_n статистически неотличимы, то они и вычислительно неотличимы, поскольку разность вероятностей попадания α_n и β_n в множество $\{x \mid C_n(x) = 1\}$, задаваемое тестом C_n , не превосходит статистического расстояния между α_n и β_n .

Нам понадобятся некоторые простые свойства понятия вычислительной неотличимости.

Лемма 3. 1) *Отношение вычислительной неотличимости рефлексивно, симметрично и транзитивно.*

2) *Пусть α_n и β_n вычислительно неотличимы, а $l(n)$ некоторая функция натурального аргумента, не превосходящая длины значений α_n . Тогда случайные величины, получаемые из α_n и β_n отрезанием последних $l(n)$ битов, вычислительно неотличимы.*

3) *Если α_n и β_n вычислительно неотличимы, p некоторый полином, а γ_n случайная величина, независимая от α_n и β_n со значениями во множестве слов длины $p(n)$, то случайные величины $\alpha_n \gamma_n$ и $\beta_n \gamma_n$ вычислительно неотличимы.*

4) *(Обобщение предыдущего.) Если имеются возможно зависимые случайные величины α_n , β_n и γ_n , причем α_n и β_n равномерно вычислительно неотличимы при любом значении γ_n , то $\alpha_n \gamma_n$ и $\beta_n \gamma_n$ вычислительно неотличимы. Равномерная неотличимость означает, что для любой последовательности значений c_n случайной величины γ_n случайные величины $(\alpha_n | \gamma_n = c_n)$ и $(\beta_n | \gamma_n = c_n)$ вычислительно неотличимы.¹*

5) *Если α_n и β_n вычислительно неотличимы, а C_n последовательность схем полиномиального от n размера с $l(n)$ входами, то случайные величины $C_n(\alpha_n)$ и $C_n(\beta_n)$ вычислительно неотличимы.*

Доказательство. Первое и второе утверждения очевидны. Докажем третье. Допустим, что существует последовательность тестов T_n полиноми-

¹ $(\alpha_n | \gamma_n = c_n)$ обозначает случайную величину с тем же множеством значений, что и у случайной величины α_n , которая принимает значение a с вероятностью $\text{Pr}[\alpha_n | \gamma_n = c_n]$.

ального от n размера с $k(n) + l(n)$ входами такая, что вероятности событий $T_n(\alpha_n \gamma_n) = 1$ и $T_n(\beta_n \gamma_n) = 1$ отличаются на $\varepsilon = 1/\text{poly}(n)$ для бесконечно многих n . Фиксируем одно из этих n . Вероятность события $T_n(\alpha_n \gamma_n) = 1$ есть среднее вероятностей событий $T_n(\alpha_n r) = 1$ для различных r из множества значений γ_n . Аналогичное верно для вероятности события $T_n(\beta_n \gamma_n) = 1$. По линейности разность средних значений любых двух случайных величины совпадает со средним значением их разности. Поэтому среднее значение разности вероятности событий $T_n(\alpha_n r) = 1$ и $T_n(\beta_n r) = 1$ не меньше ε , а значит найдется такое $r = r_n$, что вероятности событий $T_n(\alpha_n r) = 1$ и $T_n(\beta_n r) = 1$ отличаются не менее, чем на ε . Преобразование $x \mapsto T_n(xr_n)$ вычисляется некоторой схемой C_n , размер которой не больше размера T_n (“запаиваем” r_n в схему T_n). Получили противоречие с вычислительной неотличимостью α_n и β_n .

Четвертое свойство доказывается аналогично третьему: Допустим, что существует последовательность схем-тестов T_n полиномиального от n размера такая, что вероятности событий $T_n(\alpha_n \gamma_n) = 1$ и $T_n(\beta_n \gamma_n) = 1$ отличаются на $\varepsilon = 1/\text{poly}(n)$ для бесконечно многих n . Разность вероятностей событий $T_n(\alpha_n \gamma_n) = 1$ и $T_n(\beta_n \gamma_n) = 1$ равна среднему значению разности вероятностей

$$\Pr[T_n(\alpha_n c) = 1 | \gamma_n = c] - \Pr[T_n(\beta_n c) = 1 | \gamma_n = c]$$

по случайно выбранному c (в соответствии с распределением случайной величины γ_n). Поэтому для бесконечно многих n найдется $c = c_n$ в множестве значений γ_n , для которого эта разность не меньше ε по абсолютной величине. Запаивая значение c_n в схему T_n , мы получим отличитель случайных величин $(\alpha_n | \gamma_n = c_n)$ и $(\beta_n | \gamma_n = c_n)$, что противоречит условию.

Докажем пятое свойство. Пусть дана последовательность тестов T_n полиномиального от n размера с $s(n)$ входами (где $s(n)$ — количество выходов C_n) на отличимость $C_n(\alpha_n)$ и $C_n(\beta_n)$. Тогда последовательность схем $D_n(x) = T_n(C_n(x))$ можно рассматривать, как тест на отличимость α_n и β_n . Размер схемы D_n есть сумма размеров T_n и C_n , а значит ограничен полиномом от n . Следовательно вероятность того, что α_n проходит тест D_n пренебрежимо мало отличается от вероятности того, что β_n проходит тест D_n . Осталось заметить, что первая вероятность равна вероятности того, что случайная величина $C_n(\alpha_n)$ пройдет тест T_n , и то же самое верно для $C_n(\beta_n)$. \square

В некоторых приложениях нам понадобится понятие семейства попарно неотличимых величин. Пусть для каждого натурального n задано множество слов D_n полиномиальной от n длины и для каждого $u \in D_n$ задана случайная величина α_n^u со значениями в некотором множестве слов полиномиальной от n длины (эта длина зависит только от n). Мы говорим, что случайные величины α_n^u попарно вычислительно неотличимы, если для любой последовательности пар слов $u_n, v_n \in D_n$ случайные величины $\alpha_n^{u_n}$ и $\alpha_n^{v_n}$ вычислительно неотличимы.

Лемма 4. Случайные величины α_n^u попарно вычислительно неотличимы тогда и только тогда, когда любой последовательности схем T_n полиномиального размера найдется пренебрежимо малая последовательность ε_n для которой

$$|\Pr[T_n(\alpha_n^u) = 1] - \Pr[T_n(\alpha_n^v) = 1]| < \varepsilon_n$$

для всех $u, v \in D_n$.

Доказательство. Утверждение “тогда” следует непосредственно из определения. Докажем утверждение “только тогда”. Допустим случайные величины α_n^u попарно вычислительно неотличимы. Зафиксируем произвольную последовательности схем T_n полиномиального размера. Для каждого n рассмотрим пару слов $u_n, v_n \in D_n$ для которых разность

$$\Pr[T_n(\alpha_n^{u_n}) = 1] - \Pr[T_n(\alpha_n^{v_n}) = 1]$$

максимальна по абсолютной величине. Обозначим через ε_n абсолютную величину этой разности для этих u_n, v_n . По условию ε_n пренебрежимо мало. \square

Задача 23. Докажите, что определение обобщенно необратимой функции не изменится, если потребовать, чтобы трудная случайная была вычислительно (а не статистически) неотличима от некоторой случайной величины, генерируемой за полиномиальное время. Докажите, что оно также не изменится, если потребовать, чтобы трудная случайная сама генерировалась за полиномиальное время.

2.2 Определение генератора ПСЧ

Пусть даны многочлены $k(n), l(n)$ такие, что $l(n) > k(n)$ для всех n . Генератором ПСЧ типа $k(n) \rightarrow l(n)$ будем называть семейство функций

G_n , где для каждого n функция G_n отображает двоичные слова длины $k(n)$ в слова длины $l(n)$, удовлетворяющее следующим двум условиям:

(1) Семейство G_n вычислимо за полиномиальное время (по данным n и s за полиномиальное от n количество шагов можно найти $G_n(s)$).

(2) Случайная величина $G_n(s)$ (где s выбирается случайно среди всех строк длины $k(n)$) вычислительно неотличима от равномерно распределенной среди слов длины $l(n)$ случайной величины (при стремлении n к бесконечности). Это свойство генератора называют надежностью.

Задача 24. Докажите, что второе требование нельзя усилить, потребовав статистической неотличимости $G_n(s)$ и равномерно распределенной случайной величины.

Определим также генераторы ПСЧ типа $k(n) \rightarrow \infty$, как семейства G_n , где G_n отображает двоичные слова длины $k(n)$ в бесконечные двоичные последовательности, удовлетворяющие следующим требованиям.

(1) Существует алгоритм, который по слову s и натуральному l за полиномиальное от $|s| + l$ время вычисляет l -ый бит последовательности $G_n(s)$.

(2) Случайная величина $G_n(s)$ вычислительно неотличима от равномерно распределенной бесконечной последовательности нулей и единиц.

В этом определении вычислительная неотличимость двух случайных величин ω_n, ξ_n со значениями в множестве всех бесконечных двоичных последовательностей понимается в следующем смысле: для любой последовательности схем C_n размера $\text{poly}(n)$ вероятность событий $C_n(\omega_n) = 1$ и $C_n(\xi_n) = 1$ отличаются на пренебрежимо малую величину. Здесь $C_n(\omega_n)$ обозначает результат применения схемы к k первым битам ω_n , где k — количество входов C_n . Аналогично понимается $C_n(\xi_n)$. Другими словами, ω_n и ξ_n вычислительно неотличимы, если для любого полинома $p(n)$ вычислительно неотличимы случайные величины, равные и первым $p(n)$ битам случайных величин ω_n и ξ_n .

Имеется следующее соотношение между генераторами типа $k(n) \rightarrow l(n)$ и генераторами типа $k(n) \rightarrow \infty$. Если G_n — генератор типа $k(n) \rightarrow \infty$, то для любого полинома $l(n)$ последовательность функций $H_n(s) = (G_n(s))_{l(n)}$, где $s \in \{0, 1\}^{k(n)}$, будет генератором типа $k(n) \rightarrow l(n)$ (что очевидно). Обратно, мы докажем, что по любому генератору типа $k(n) \rightarrow l(n)$ (какие бы ни были $k(n) < l(n)$) можно построить генератор типа $k(n) \rightarrow \infty$.

Задача 25. Докажите, что существует функция G_n (не обязательно вы-

числимая за полиномиальное время), типа $n \rightarrow n + 1$ такая, что случайная величина $G_n(s)$ вычислительно неотличима от равномерно распределенной случайной величины.

Задача 26. Докажите, что если функция G_n является генератором типа $k(n) \rightarrow l(n)$, а x_n последовательность слов длины $l(n)$, вычислимая за время $\text{poly}(n)$, то функция $s \mapsto G_n(s) \oplus x_n$ является генератором.

2.3 Генераторы ПСЧ и односторонние функции

Любой генератор G_n типа $k(n) \rightarrow l(n)$ является слабо необратимой функцией. Действительно, никакая последовательность схем C_n полиномиального размера не может обратить $G_n(s)$ в вероятность успеха, большей $3/4$ (для бесконечно многих n). Допустим, такая последовательность схем существует. Тогда рассмотрим следующий тест на последовательностях длины $l(n)$: применяем к последовательности y схему C_n , затем проверяем, с помощью алгоритма, вычисляющего G_n , правильно ли схема C_n обратила y (то есть, $G_n(C_n(y)) = y$). Если обращение произошло удачно, то выдаем 1, а иначе 0. Вероятность того, что тест будет пройден случайной величиной $G_n(s)$, не менее $3/4$ (для бесконечно многих n). Вероятность того, что равномерно распределенная случайная величина пройдет тест, не больше $1/2$, поскольку множество значений G_n составляет не более половины всего множества последовательностей длины $l(n)$. Поскольку C_n и G_n можно вычислить схемой полиномиального от n размера, описанный тест имеет полиномиальный размер.

Итак, если для каких-то $l(n) > k(n)$ существует генератор типа $k(n) \rightarrow l(n)$, то существуют и слабо односторонние, а значит и сильно односторонние функции. Оказывается, верно и обратное.

Теорема 5. [2] *Если существует сильно односторонняя функция, то существует и генератор ПСЧ типа $n \rightarrow \infty$.*

Мы докажем более слабое утверждение: если существует односторонняя перестановка, то существует и генератор ПСЧ типа $n \rightarrow \infty$. Сначала дадим определение односторонней перестановки.

Функция $f_n : D_n \rightarrow D_n$ называется односторонней перестановкой, если (1) $D_n \subset \{0, 1\}^{k(n)}$ и отображение $n, x \mapsto f_n(x)$ вычислимо за время $\text{poly}(n)$, (2) функция f_n является перестановкой множества D_n и (3) случайная величина, равномерно распределенная на D_n является доступной и трудной для f_n .

Приведем примеры (предположительно) односторонних перестановок. Известны три таких примера. А именно, в областях определения функции Рабина, функции RSA и функции Блюма-Микэли мы выделим некоторое подмножество D_n с такими свойствами: (1) равномерное распределение на D_n доступно и (2) представляется правдоподобным, что равномерное распределение на D_n является трудным (говоря точнее, до сих пор не удалось опровергнуть это предположение).

Функция Рабина. Рассмотрим следующее множество D_n . Оно состоит из всех строк вида xy , где $y = p \cdot q$ и p, q взаимно простые n -битовые числа вида $4k + 3$, причем x и y взаимно просты, а $x \in [0, y)$ является полным квадратом по модулю y . Докажем, что функция Рабина является перестановкой D_n . Ясно, что f отображает D_n в себя. Поэтому достаточно доказать, что f инъективна. Пусть $x = z^2$ и z взаимно просто с p . Тогда по x^2 можно найти x по модулю p , возведя x в степень $(p + 1)/4$. Действительно, $(x^2)^{(p+1)/4} = z^{p+1} = z^2 = x \pmod{p}$. То же самое верно и по модулю q , поэтому по x^2 и $y = pq$ можно восстановить x . (Из этого, конечно, не следует обратимость функции Рабина, поскольку для ее обращения нужно разложить y на простые множители.)

Статистически неотличимую от равномерно распределенной на D_n величину можно генерировать следующим образом так же, как раньше. При этом нам важно, что простых чисел вида $4k + 3$ достаточно много: среди всех n -битовых чисел они составляют долю не менее $1/\text{poly}(n)$.

Функция RSA. Для функции RSA рассмотрим следующее подмножество D_n ее области определения: в него включаются только те xuz , для которых y есть произведение двух простых n -битовых чисел p и q , число x находится в интервале $[0, pq)$, а z взаимно просто со значением функции Эйлера на pq , $\phi(pq) = (p - 1)(q - 1)$. Очевидно, что функция RSA отображает D_n в себя. Инъективность следует из того, что по модулю $\phi(pq)$ число z имеет обратный элемент u . Поэтому по x^z можно найти x , возведя x^z в степень u . Действительно, $\phi(pq)$ есть мощность мультипликативной группы вычетов по модулю pq и по теореме Лагранжа мы имеем $x^{zu} = x^{1+k\phi(pq)} = x \pmod{pq}$. Из этого, конечно не следует, обратимость функции RSA, поскольку для обращения нужно знать u (или числа p и q , по которым u можно вычислить).

Доступность равномерного распределения на D_n не очевидна из-за того, что надо уметь порождать числа, взаимно простые с $(p - 1)(q - 1)$. Для этого нам нужно знать разложение $(p - 1)(q - 1)$ на простые множители. Оказывается, можно порождать равномерное распределение на

простых n -битовых числах вместе с разложением числа $p - 1$ на простые множители (см. [1]).

Функция Блюма и Микэли. Для функции Блюма и Микэли рассмотрим следующее подмножество D_n ее области определения: в него включаются только те xuz , y в для которых y есть n -битовое простое число, x является генератором мультипликативной группы вычетов по модулю y , а $z \in [0, y - 1)$. По определению генератора, функция Блюма и Микэли является перестановкой D_n . Доступность равномерного распределения на D_n неочевидна, это доказано в [1].

Теорема 6. *Если существует односторонняя обобщенная перестановка, то существует генератор типа $n \rightarrow \infty$.*

2.4 Трудный бит

Сначала мы научимся добавлять один случайный бит к уже имеющимся. Для этого нам понадобится понятие трудного бита для данной функции. Пусть β_n, γ_n последовательность пар совместно распределенных случайных величин, причем случайная величина β_n имеет значения 0,1. Мы говорим, что β_n является трудно вычисляемой по γ_n если для любой последовательности схем C_n размера $\text{poly}(n)$ вероятность события $C_n(\gamma_n) = \beta_n$ приблизительно равна $1/2$ (то есть, стремится к $1/2$ быстрее любого обратного полинома от n). То есть, случайную величину β_n нельзя вычислить по γ_n с вероятностью существенно лучшей, чем при простом угадывании.² Если β_n является трудно вычисляемой по γ_n , то оба своих значения она принимает с примерно равной вероятностью (иначе в качестве C_n можно взять схему, которая выдает 0 независимо от входа).

Определение. Полиномиально вычисляемая функция $h_n : D_n \rightarrow \{0, 1\}$ является трудным битом для функции $g_n : D_n \rightarrow D_n$, если случайная величина $h_n(\alpha_n)$ трудно вычислима по случайной величине $g_n(\alpha_n)$, где α_n — случайная величина, равномерно распределенная в D_n .

Задача 27. Докажите, что если имеется трудный бит для перестановки $g_n : D_n \rightarrow D_n$, то перестановка сильно необратима.

Трудным битом для функции Рабина является последний бит (четность), в предположении необратимости функции Рабина. Трудным битом для функции Блюма–Микэли (в предположении необратимости этой

²В общем случае вероятность должна быть приблизительно равной $1/|M_n|$, где M_n — множество возможных значений β_n .

функции) является функция $h_n(xyz) = 0$, если $z < (y - 1)/2$, и $h_n(xyz) = 1$ иначе.

Лемма 7. *Случайная величина β_n трудно вычислима по γ_n тогда и только тогда, когда случайные величины $\beta_n\gamma_n$ и $r\gamma_n$ вычислительно неотличимы. (Здесь r есть равномерно распределенный бит, независимый от γ_n .)*

Доказательство. В одну сторону утверждение леммы очевидно: если $\beta\gamma$ и $r\gamma$ вычислительно неотличимы, то β является трудным битом для γ . (Мы опускаем индекс n .) Действительно, пусть имеется последовательность схем C_n полиномиальной от n размера такая, что для бесконечно многих n вероятность события $C(\gamma) = \beta$ отличается от $1/2$ не менее, чем на $\varepsilon = 1/\text{poly}(n)$. Рассмотрим следующий тест D : он применяет ко входной последовательности без первого бита C и выдает 1, если результат равен первому биту входной последовательности. Случайная величина $r\gamma$ проходит этот тест с вероятностью ровно $1/2$, поскольку ее первый бит не зависит от последующих битов и принимает значения 0,1 с равными вероятностями. Вероятность того, что последовательность $\beta\gamma$ пройдет этот тест, в точности равна вероятности того, что $C(\gamma) = \beta$. Ясно, что тест D задается схемой полиномиальной от n размера.

Теперь докажем обратное. Допустим, что существует последовательность схем D размера $\text{poly}(n)$ такая, что для бесконечно многих n выполнено

$$\Pr[D(\beta\gamma) = 1] - \Pr[D(r\gamma) = 1] = \varepsilon,$$

где $|\varepsilon| \geq 1/\text{poly}(n)$. Фиксируем любое из этих бесконечно многих n . Пусть для определенности $\varepsilon > 0$. Это значит, что схема D “больше любит” β , чем случайный бит. Рассмотрим следующий алгоритм C вычисления β по γ .

Если $D(0\gamma) = D(1\gamma)$, то выдаем 0 и 1 с равными вероятностями. Если $D(0\gamma) = 0$, $D(1\gamma) = 1$, выдаем 1 (неформальное объяснение: β скорее равен 1, чем 0, поскольку схема больше любит 1, чем 0). Если $D(0\gamma) = 1$, $D(1\gamma) = 0$, выдаем 0.

Вероятность того, что этот алгоритм выдаст β равна $1/2 + \varepsilon$. Чтобы доказать это, достаточно доказать это при любом фиксированном значении $\bar{\gamma}$ случайной величины γ выполнено равенство

$$\Pr[C(\bar{\gamma}) = \beta] = 1/2 + \Pr[D(\beta\bar{\gamma}) = 1] - \Pr[D(r\bar{\gamma}) = 1]. \quad (1)$$

(В этом уравнении все вероятности вычисляются при условии $\gamma = \bar{\gamma}$.) Пусть $\bar{\gamma}$ фиксировано. Если $D(0\bar{\gamma}) = D(1\bar{\gamma})$ то схема не чувствует замены 0 на 1. Поэтому правая часть (1) равна $1/2$. Левая часть также равна $1/2$ по определению C .

Если $D(0\bar{\gamma}) = 0$, $D(1\bar{\gamma}) = 1$, то левая часть (1) равна $\Pr[1 = \beta]$, а в правая равна $1/2 + \Pr[\beta = 1] - 1/2$, и равенство выполнено. Если $D(0\bar{\gamma}) = 1$, $D(1\bar{\gamma}) = 0$, то левая часть (1) равна $\Pr[0 = \beta]$, а правая равна $1/2 + \Pr[\beta = 0] - 1/2$.

Алгоритм C задается вероятностной схемой примерно вдвое большей, чем D , поэтому имеющей также полиномиальный размер. Можно обойтись и схемой размера, примерно равного размеру схемы D , если заметить, что $C(\gamma) = D(\gamma r) \oplus r \oplus 1$, где r случайный бит, независимый от γ . \square

Пусть h_n — трудный бит для односторонней перестановки $g_n : D_n \rightarrow D_n$, а случайная величина α_n равномерно распределена в D_n . По лемме случайные величины $h_n(\alpha_n)g_n(\alpha_n)$ и $rg_n(\alpha_n)$ вычислительно неотличимы. Вторая имеет то же распределение, что и $r\alpha_n$. Таким образом, случайные величины $h_n(\alpha_n)g_n(\alpha_n)$ и $r\alpha_n$ вычислительно неотличимы. Говоря неформально, случайная величина $h_n(\alpha_n)g_n(\alpha_n)$ имеет на один бит случайности больше, чем α_n . Повторяя этот прием, можно добавить второй, третий случайные биты, и так далее. На этом приеме и основано построение генераторов ПСЧ.

Пусть дан произвольный полином $p(n)$, задающий количество случайных битов, которые нам нужно получить. Рассмотрим последовательность

$$\alpha, g(\alpha), g(g(\alpha)), \dots, g^{p(n)-1}(\alpha)$$

(для наглядности мы опускаем индекс n). Применим ко всем членам этой последовательности функцию h . Утверждается, что полученная последовательность

$$h(\alpha), h(g(\alpha)), h(g(g(\alpha))), \dots, h(g^{p(n)-1}(\alpha))$$

будет вычислительно неотличима от случайной равномерно распределенной последовательности (при $n \rightarrow \infty$). На самом деле, мы докажем чуть больше: если добавить в хвост этой последовательности, $g^{p(n)}(\alpha)$, то полученная последовательность будет вычислительно неотличима от равномерно распределенной последовательности длины $p(n)$ с приписанным в конец α_n . Это усиление нам понадобится в дальнейшем.

Лемма 8. Пусть h_n является трудным битом для перестановки $g_n : D_n \rightarrow D_n$. Тогда для любого полинома $p(n)$ случайные величины

$$h(\alpha), h(g(\alpha)), \dots, h(g^{p(n)-1}(\alpha)), g^{p(n)}(\alpha) \quad \text{и} \quad r_1, r_2, \dots, r_{p(n)}, \alpha$$

вычислительно неотличимы (для наглядности мы опускаем индекс n). Здесь α равномерно распределена в D_n , а $r_1 r_2 \dots r_{p(n)}$ случайная равномерно распределенная строка длины $p(n)$, независимая от α .

Доказательство. Сначала докажем утверждение для случаев $p(n) \equiv 1$ и $p(n) \equiv 2$. В первом случае вычислительная неотличимость случайных величин $h(\alpha)g(\alpha)$ и $r_1\alpha$ следует из Леммы 7: $h(\alpha)g(\alpha) \equiv r_1g(\alpha) \equiv r_1\alpha$. Вторая из эквивалентностей имеет место, поскольку g перестановка.

Теперь докажем утверждение для $p(n) \equiv 2$. Надо установить неотличимость случайных величин $h(\alpha)h(g(\alpha))g^2(\alpha)$ и $r_1r_2\alpha$. Рассмотрим “промежуточную” случайную величину $r_1h(\alpha)g(\alpha)$.

Докажем ее неотличимость от обеих случайных величин. Неотличимость от $r_1r_2\alpha$ следует из неотличимости $h(\alpha)g(\alpha)$ и $r_2\alpha$, данной нам в условии (обе случайные величины получаются приписыванием к этим случайным величинам одного бита).

Неотличимость случайных величин $h(\alpha)h(g(\alpha))g^2(\alpha)$ и $r_1h(\alpha)g(\alpha)$ устанавливается немного сложнее. Рассмотрим преобразование T , которое переводит строку $r_1\alpha$ в строку $r_1h(\alpha)g(\alpha)$. Это преобразование вычислимо схемами полиномиального размера. С другой стороны, это же преобразование переводит случайную величину $h(\alpha)g(\alpha)$ в случайную величину $h(\alpha)h(g(\alpha))g^2(\alpha)$. Поскольку случайные величины $r_1\alpha$ и $h(\alpha)g(\alpha)$ неотличимы, будет неотличимы и результаты применения к ним преобразования T .

Теперь докажем утверждение в общем случае. Допустим последовательность схем C_n полиномиального размера отличает случайные величины

$$h(\alpha), h(g(\alpha)), \dots, h(g^{p(n)-1}(\alpha)), g^{p(n)}(\alpha) \quad \text{и} \quad r_1, r_2, \dots, r_{p(n)}, \alpha.$$

То есть, для бесконечно многих n вероятность того, что первая случайная величина проходит тест C_n , отличается не менее, чем на $\varepsilon_n = 1/\text{poly}(n)$, от вероятности того, что вторая случайная величина проходит тест C_n . Фиксируем любое такое n и для каждого $i = 0, \dots, p(n)$ рассмотрим промежуточные величины

$$H_i = r_1, r_2, \dots, r_i, h(\alpha), h(g(\alpha)), \dots, h(g^{p(n)-1-i}(\alpha)), g^{p(n)-i}(\alpha).$$

Случайные величины H_0 и $H_{p(n)}$ совпадают со случайными величинами из условия леммы. Поэтому для некоторого i будет выполнено

$$|\Pr[C_n(H_i) = 1] - \Pr[C_n(H_{i+1}) = 1]| \geq \varepsilon_n/p(n).$$

Случайная величина H_{i+1} получена некоторым преобразованием T строки $r_1r_2 \dots r_i r_{i+1}\alpha$, вычисляемым схемой размера $\text{poly}(n)$. Если это преобразование применить к строке $r_1r_2 \dots r_i h(\alpha)g(\alpha)$ то получится случайная величина H_i . Поэтому схема $C_n(T(r_1r_2 \dots r_i r_{i+1}\alpha))$ различает величины $r_1r_2 \dots r_i r_{i+1}\alpha$ и $r_1r_2 \dots r_i h(\alpha)g(\alpha)$, которые неотличимы, поскольку таковы $r_{i+1}\alpha$ и $h(\alpha)g(\alpha)$. \square

Итак, мы почти доказали следующее утверждение.

Теорема 9. *Если существует односторонняя перестановка и трудный бит для нее, то для некоторого полинома $q(n)$ существует генератор ПСЧ типа $q(n) \rightarrow \infty$.*

Доказательство. Сначала предположим, что D_n есть множество всех слов длины $k(n)$. Тогда в качестве генератора $q(n)$ можно взять $k(n)$ и положить

$$G_n(s) = h(s)h(g(s))h(g^2(s)) \dots$$

По Лемме 8 это отображение является генератором ПСЧ. Действительно, лемма утверждает, что любое начало этой последовательности полиномиальной длины вычислительно неотлично от равномерно распределенной последовательности той же длины.

В общем случае воспользуемся доступностью равномерного распределения на D_n . Рафиксируем вероятностный алгоритм K , который по n за полиномиальное от n время генерирует случайную величину, которая статистически неотличима от случайной величины α_n , равномерно распределенной в D_n . Пусть K использует случайную строку s длины $q(n)$ и $K_n(s)$ обозначает выдаваемую им строку. Определим генератор G_n . Его значение на строке s длины $q(n)$ равно

$$G_n(s) = h(K_n(s))h(g(K_n(s)))h(g^2(K_n(s))) \dots$$

В этой последовательности l -ый бит равен $h(g^l(K_n(s)))$ и может быть вычислен за полиномиальное от $l + n$ время.

Докажем, что построенный генератор G_n надежен. Пусть $p(n)$ — произвольный полином. Случайная величина $K_n(s)$ статистически неотличима от случайной величины α_n . Поэтому и первые $p(n)$ битов $G_n(s)$ неотличимы от

$$h(\alpha_n)h(g(\alpha_n))h(g^2(\alpha_n)) \dots h(g^{p(n)-1}(\alpha_n))$$

тестами того же размера.³ По Лемме 8 последняя неотличима от равномерно распределенной последовательности тестами размера $\text{poly}(n)$. \square

Генератор, построенный в доказательстве теоремы 9 обладает двумя практически важными свойствами, не отраженным в определении. (1) Биты последовательности $G_n(s)$ можно вычислять по очереди так, что на вычисление очередного бита уходит время, ограниченное некоторым полиномом от n , не зависящим от номера этого бита. Действительно, вычислив l первых битов $G_n(s)$, мы сохраняем слово $g^l(s)$. Для вычисления очередного бита надо применить к $g^l(s)$ сначала функцию g , а потом к полученному слову применить функцию h . (2) Для любого полинома p любые $p(n)$ подряд идущих битов (а не только первые) сгенерированной последовательности вычислительно неотличимы от равномерно распределенной последовательности длины $p(n)$. Действительно, случайная величина $g^l(s)$ имеет в точности то же распределение, что и s . Поэтому подслово $G_n(s)$, состоящее из битов с номерами $l, l+1, \dots, l+p(n)-1$, имеет в точности то же распределение, что и начало $G_n(s)$ длины $p(n)$.

Замечание. В каком месте доказательства Леммы 8 мы использовали то, что g_n является перестановкой? Ровно в одном месте, а именно, когда использовали вычислительную неотличимость случайных величин α_n и $g_n(\alpha_n)$. Поэтому в условии леммы требование инъективности g_n можно заменить на более слабое требование: случайные величины α_n и $g_n(\alpha_n)$ вычислительно неотличимы (где α_n равномерно распределена в D_n). Более того, совершенно необязательно, чтобы случайная величина α_n была равномерно распределена в D_n . Нам нужно, чтобы она была доступна и трудна для g_n . Будем функции g_n , для которых существуют доступная трудная случайная величина α_n , вычислительно неотличимая от $g_n(\alpha_n)$, называть *обобщенными перестановками*.

Итак, мы получаем следующие утверждения.

³В этом доказательстве нам достаточно всего лишь вычислительной неотличимости $K_n(s)$ и α_n .

Лемма 10. Пусть функции g_n, h_n вычислимы за полиномиальное время, причем h_n принимает значения $0, 1$. Пусть случайная величина $h_n(\alpha_n)$ трудно вычислима по $g_n(\alpha_n)$, где α_n доступная случайная величина, вычислительно неотличимая от $g_n(\alpha_n)$. Тогда для любого полинома $p(n)$ случайные величины

$$h(\alpha), h(g(\alpha)), \dots, h(g^{p(n)-1}(\alpha)), g^{p(n)}(\alpha) \quad \text{и} \quad r_1, r_2, \dots, r_{p(n)}, \alpha$$

вычислительно неотличимы (для наглядности мы опускаем индекс n). Здесь $r_1 r_2 \dots r_{p(n)}$ случайная равномерно распределенная строка длины $p(n)$, независимая от α_n .

Теорема 11. Если существуют полиномиально вычисляемые функции g_n, h_n и доступная случайная величина α_n такие, что $h_n(\alpha_n)$ трудно вычислимо по $g_n(\alpha_n)$ и α_n вычислительно неотличимо от $g_n(\alpha_n)$, то для некоторого полинома $q(n)$ существует генератор ПСЧ типа $q(n) \rightarrow \infty$.

Из этой теоремы мы получаем следующее следствие:

Теорема 12. Если для некоторой полиномиально вычисляемой функции $k(n)$, ограниченной сверху полиномом, существует генератор G_n типа $k(n) \rightarrow k(n) + 1$, то для некоторого полинома $q(n)$ существует генератор ПСЧ типа $q(n) \rightarrow \infty$.

Доказательство. Обозначим через $h_n(s)$ первый бит $G_n(s)$, а через $g_n(s)$ остальные $k(n)$ битов. Мы получили полиномиально вычисляемую обобщенную перестановку $g_n : \{0, 1\}^{k(n)} \rightarrow \{0, 1\}^{k(n)}$ вместе с трудным битом для нее. Осталось воспользоваться Теоремой 11. \square

Задача 28. Докажите, что если для некоторого полинома $q(n)$ существует генератор ПСЧ типа $q(n) \rightarrow \infty$, то существует и генератор ПСЧ типа $n \rightarrow \infty$.

2.5 Отступление: надежность генераторов по Яо

В некоторых приложениях генераторов псевдослучайных чисел нам всего лишь нужно, чтобы невозможно было предсказывать никакой бит сгенерированной последовательности по ее предыдущим битам с вероятностью, существенно отличающейся от $1/2$ (например, в казино). Генераторы, удовлетворяющие этому требованию называются надежными по Яо (Yao).

Определение. Пусть имеется последовательность случайных величин $\alpha^1, \alpha^2, \alpha^3, \dots$ и каждая α^n принимает значения в множестве бесконечных последовательностей нулей и единиц. Мы будем говорить, что α^n неотличима по Яо от равномерно распределенной последовательности, если для любой последовательности индексов $i_n \leq \text{poly}(n)$ случайная величина $\alpha^n_{i_n+1}$ ($i_n + 1$ -ый бит α^n) является трудно вычислима по случайной величине $(\alpha^n)_{i_n}$ (начало длины i_n последовательности α^n).⁴ Генератор типа $n \rightarrow \infty$ называется *надежным по Яо*, если генерируемая им случайная величина неотличима по Яо от равномерно распределенной последовательности.⁵

Если случайная величина вычислительно неотличима от равномерно распределенной случайной величины, то она неотличима от нее и по Яо. Действительно, по лемме 7 случайная величина $(\alpha^n)_{i_n+1}$ трудно вычислима по случайной величине $(\alpha^n)_{i_n}$ тогда и только тогда, когда случайные величины $(\alpha^n)_{i_n+1}$ и $(\alpha^n)_{i_n}$ вычислительно неотличимы. А последнее верно, поскольку обе случайные величины неотличимы от случайной величины, от равномерно распределенной среди слов длины $i_n + 1$. Верно и обратное.

Теорема 13. *Если случайная величина α^n в множестве бесконечных последовательностей нулей и единиц неотличима по Яо от равномерно распределенной случайной величины, то она вычислительно неотличима от нее. Следовательно, любой надежный по Яо генератор надежен в обычном смысле.*

Доказательство. Пусть C_n произвольная последовательность схем размера $\text{poly}(n)$. Нам надо доказать, что C_n не отличает α^n от равномерно распределенной случайной величины. Обозначим через $l(n)$ количество входов C_n . Фиксируем произвольное n и для каждого $0 \leq i \leq l(n)$ рассмотрим случайную величину H_i , которая получится, если в последовательности α^n заменить все биты, начиная с $i + 1$ -ого, на случайные биты, независимые от α^n . Ясно, что начало $H_{l(n)}$ длины $l(n)$ распределено так же, как начало α^n длины $l(n)$, а H_0 равномерно распределена. Обозначим через i_n тот индекс, для которого $|\Pr[C_n(H_i) = 1] - \Pr[C_n(H_{i+1}) = 1]|$ максимально. Сравним случайные величины H_i и H_{i+1} . У них первые i

⁴Можно дать определение неотличимости по Яо любых двух случайных величин в множестве слов одной длины, но оно нам не понадобится.

⁵Аналогично дается определение надежности по Яо генераторов вида $k(n) \rightarrow l(n)$.

битов равны i первым битам α^n . Последние $n - i - 1$ битов выбираются случайно. Разница только в $i + 1$ -ом бите, который в H_i выбирается случайным образом, а в H_{i+1} берется из α^n . Поскольку α^n неотличима по Яо от равномерно распределенной последовательности, $\alpha_{i_{n+1}}$ (верхний индекс n опускаем) невозможно вычислить по $(\alpha)_{i_n}$ с вероятностью, существенно отличающейся от $1/2$. По лемме 7 случайные величины $(\alpha)_{i_{n+1}}$ и $(\alpha)_{i_n} r$ вычислительно неотличимы (здесь r случайный равномерно распределенный бит, независимый от α). Значит и случайные величины H_{i_n} и $H_{i_{n+1}}$ вычислительно неотличимы (они получаются приписыванием к $(\alpha)_{i_{n+1}}$ и $(\alpha)_{i_n} r$, соответственно, случайной независимой последовательности). Следовательно, величина $|\Pr[C_n(H_{i_n}) = 1] - \Pr[C_n(H_{i_{n+1}}) = 1]|$ стремится к нулю быстрее любого обратного полинома от n . Поскольку величина $|\Pr[H_0] = 1] - \Pr[C_n(H_{l(n)}) = 1]|$ не более, чем в $l(n)$ раз превосходит эту, то и она стремится к нулю быстрее любого обратного полинома от n . \square

2.6 Построение функции, являющейся трудным битом

Пусть $f_n : D_n \rightarrow D_n$ односторонняя перестановка, и $D_n \subset \{0, 1\}^{l(n)}$. Рассмотрим функции g_n, h_n определенные на множестве $D_n \times \{0, 1\}^{l(n)}$ равенствами $g_n(xy) = f_n(x)y$ и $h_n(xy) = x \odot y$. Здесь выражение xy обозначает конкатенацию слов x, y длины $l(n)$, а $x \odot y = \sum_{i=1}^{l(n)} x_i y_i \bmod 2$. Ясно, что функции g_n и h_n полиномиально вычислимы и g_n является перестановкой. Теорема Гольдрайха—Левина утверждает, что h_n является трудным битом для нее.

Теорема 14 (Гольдрайха—Левина). *Если f_n сильно необратима, то функция $x \odot y$ является трудным битом для функции $g_n(x, y) = f_n(x)y$.*

Доказательство. Доказательство основано на возможности декодирования списком кодов Адамара. Кодом Адамара слова x длины m называется последовательность длины 2^m , состоящая из значений $x \odot y$ для всех слов y длины m . Ясно, что по коду Адамара можно восстановить само слово — i -ый бит x есть скалярное произведение x с i -ым единичным вектором e_i . Нам понадобится способность кода Адамара “исправлять ошибки”. Это означает, что если в коде Адамара слова x изменить не более чем $(1/2 - \varepsilon)2^m$ битов, то по полученной строке можно отыскать список из $\text{poly}(m/\varepsilon)$ строк, среди которых есть x .

Лемма 15. *Существует вероятностный алгоритм, который, для любого слова x длины t и любого ε , получив на вход t и ε , и имея в качестве оракула любую функцию $s(y)$ такую, что $\Pr[s(y) = x \odot y] \geq 1/2 + \varepsilon$, за полиномиальное от $t, 1/\varepsilon$ время с вероятностью не менее $1/2$ находит некоторый список слов (размера $\text{poly}(t, 1/\varepsilon)$), содержащий x .*

Эту лемму мы докажем позже. А сейчас завершим доказательство теоремы, используя ее.

Допустим противное — для бесконечно многих n существует схема C_n размера $\text{poly}(n)$ такая, что вероятность события $C_n(f(x)y) = x \odot y$ не меньше $1/2 + \varepsilon_n$, где $\varepsilon_n = 1/\text{poly}(n)$ (слово x выбирается равномерно из D_n , слово y выбирается равномерно из $\{0, 1\}^{l(n)}$ независимо от x). Зафиксируем некоторое n из этого бесконечного множества и построим вероятностную схему полиномиального от n размера, которая будет обрабатывать $f(x)$ с вероятностью не меньше $\varepsilon_n/4$. Эта схема работает так: она запускает алгоритм из леммы 15, используя в качестве внешней процедуры $s(y) = C_n(f(x)y)$, и $\varepsilon_n/2$ в качестве ε . Затем для каждого из \tilde{x} , выданных алгоритмом, схема вычисляет $f(\tilde{x})$, сравнивая полученное значение с $f(x)$ и выдает первое \tilde{x} , для которого $f(\tilde{x}) = f(x)$. Если такого \tilde{x} в списке не нашлось, то схема выдает, что угодно.

Оценим вероятность с которой эта схема правильно обрабатывает $f(x)$. Обозначим через $p(x)$ вероятность события $C_n(f(x)y) = x \odot y$. По лемме 15 схема обратит $f(x)$ с вероятностью не меньше $1/2$ для любого такого x , что $p(x) \geq 1/2 + \varepsilon_n/2$. Поэтому достаточно доказать, что с вероятностью не меньше $\varepsilon_n/2$ выполнено $p(x) \geq 1/2 + \varepsilon_n/2$. Это легко доказать от противного. Действительно, по условию среднее значение величины $p(x)$ не меньше $1/2 + \varepsilon_n$. Разделим все x на те, для которых $p(x) \geq 1/2 + \varepsilon_n/2$, и остальные. Если бы x попадало в первый класс с вероятностью меньше $\varepsilon_n/2$, то среднее значение величины $p(x)$ было бы меньше $(\varepsilon_n/2) \cdot 1 + 1 \cdot (1/2 + \varepsilon_n/2) = 1/2 + \varepsilon_n$ (первое слагаемое в этой сумме ограничивает сверху вклад в среднее всех x из первой группы, второе слагаемое — вклад всех остальных). Теорема доказана (по модулю Леммы 15). \square

Замечание. В этом доказательстве мы не использовали то, что трудное распределение для g_n является равномерным. Таким образом, верно следующее обобщение теоремы Голдрайха–Левина и ее следствие.

Теорема 16. Пусть α_n доступная трудная случайная величина для функции g_n . Тогда случайная величина $x \odot y$ трудно вычислима по случайной величине $g_n(x)y$, где x распределено так же, как α_n , а y независимо от x и распределено равномерно среди слов той же длины, что и α_n .

Теорема 17. Если существует обобщенно односторонняя перестановка, то для некоторого полинома $q(n)$ существует генератор ПСЧ типа $q(n) \rightarrow \infty$.

Доказательство. Пусть даны перестановка g_n множества D_n и трудная для нее случайная величина α_n , вычислительно неотличимая от $g_n(\alpha_n)$. Рассмотрим функцию $(x, y) \mapsto (g_n(x), y)$, где $x \in D_n$, а y произвольное слово той же длины, что и x . Будем выбирать x случайно по распределению α_n , а y равномерно и независимо от x . Такое распределение на парах (x, y) доступно, и по предыдущей теореме $x \odot y$ трудно вычислимо по $(g_n(x), y)$. Кроме того, (x, y) вычислительно неотлично от $(g_n(x), y)$. Осталось применить Теорему 11. \square

2.7 Доказательство Леммы 15

Декодирование кода Адамара, испорченного менее чем в четверти битов

Сначала объясним, как можно за экспоненциальное от m (длины слова x) исправлять код Адамара слова x , испорченный менее чем в четверти битов. Пусть z неиспорченный код Адамара слова x , то есть слово длины 2^m , у которого бит номер y равен $x \odot y$. Пусть \tilde{z} — испорченный код Адамара, то есть некоторое слово, которое отличается от z менее, чем в четверти позиций. Нам дано \tilde{z} и надо найти x .

Будем находить биты слова x по очереди. Для того, чтобы найти i -ый бит x_i , перенумеруем мысленно биты слова z другим способом. Раньше бит номер y кода Адамара был равен $x \odot y$, а теперь он будет равен $x \odot (y \oplus e_i)$. (Напомним, что e_i обозначает слово, имеющее в i -ой позиции, и нули в остальных.) Обозначим получившееся слово через z^i . Нетрудно понять, что

$$z^i(y) = x \odot (y \oplus e_i) = (x \odot y) \oplus (x \odot e_i) = (x \odot y) \oplus x_i.$$

Поэтому, если i -ый бит слова x равен нулю, то $z^i = z$, а иначе $z^i = \bar{z}$ (так мы обозначили слово z , в котором все биты инвертированы). Обозначим через \tilde{z}^i результат аналогичной перенумерации битов слова \tilde{z} . Поскольку слово \tilde{z} отличается от слова z менее чем в четверти позиций, для него верно следующее. Если i -ый бит слова x равен нулю, то \tilde{z}^i отличается от \tilde{z} менее чем в половине позиций, а иначе они отличаются более чем в половине позиций. Поэтому, сравнив \tilde{z}^i и \tilde{z} мы сможем понять, чему равен i -ый бит x .

Декодирование списком кода Адамара, испорченного менее чем в половине битов, за экспоненциальное время

Пусть теперь нам известно, что слово \tilde{z} отличается от кода Адамара z слова x не более чем в $(1/2 - \varepsilon)2^m$ позиций. Объясним, как за экспоненциальное от m время найти список размера $\text{poly}(1/\varepsilon)$, содержащий x . Ясно, что надо найти список всех слов x' коды Адамара которых отличаются от \tilde{z} не более чем в $(1/2 - \varepsilon)2^m$ позиций (слово x содержится в этом списке, и обратно, любое слово из этого списка могло бы быть исходным x). Поэтому нам необходимо и достаточно убедиться, что этот список содержит $\text{poly}(1/\varepsilon)$ слов. Мы докажем, что в нем не более $(1/2\varepsilon)^2$ слов.

Для этого рассмотрим на линейном пространстве всех функций из $\{0, 1\}^m$ в \mathbb{R} скалярное произведение (r, t) определяемое как среднее значение случайной величины $r(y)t(y)$:

$$(r, t) = 2^{-m} \sum_y r(y)t(y).$$

(Нетрудно проверить, все аксиомы скалярного произведения выполнены.) Сделаем в коде Адамара слова x замены $1 \rightarrow -1$ и $0 \rightarrow 1$. То есть отныне будем считать, что y -ый бит кода Адамара слова x равен $(-1)^{x \odot y}$. Что можно сказать о скалярном произведении z и \tilde{z} (биты \tilde{z} заменяются аналогичным образом)? В побитовом произведении z и \tilde{z} будет не более чем $(1/2 - \varepsilon)2^m$ минус единиц, а остальные члены побитового произведения будут равны 1. Поэтому

$$(z, \tilde{z}) \geq (1/2 + \varepsilon) - (1/2 - \varepsilon) = 2\varepsilon.$$

Алгоритм из предыдущего параграфа (восстановления кода Адамара, испорченного менее чем в четверти битов) можно изложить в этих терминах так. Инвертирование менее четверти битов в одном из векторов

u, v меняет скалярное произведение (u, v) менее чем на $1/2$. Поэтому скалярное произведение (\tilde{z}, \tilde{z}^i) отличается от (z, z^i) менее чем на 1. Последнее же равно $+1$ или -1 в зависимости от x_i . Поэтому по знаку (\tilde{z}, \tilde{z}^i) можно найти x_i .

Нетрудно проверить, что коды Адамара всевозможных слов длины m образуют ортонормированный базис в пространстве \mathbb{R}^{2^m} : они попарно ортогональны, скалярный квадрат каждого из них равен 1, а их количество равно размерности пространства. Поэтому для любой функции r сумма квадратов скалярных произведений r с этими функциями равна скалярному квадрату r . В частности, это верно и для \tilde{z} :

$$\sum_z (\tilde{z}, z)^2 = (\tilde{z}, \tilde{z})^2 = 1.$$

(Здесь z пробегает коды Адамара всех строк длины m .) Следовательно, количество z , для которых $(z, \tilde{z}) \geq 2\varepsilon$, не превосходит $1/(2\varepsilon)^2$. Все такие z можно найти перебором за время $2^{O(m)}$ и среди них присутствует код Адамара исходной строки.

Полиномиальное декодирование списком кода Адамара, испорченного менее чем в половине битов

В доказательстве нам понадобится способ “дешево” производить много попарно независимых равномерно распределенных векторов длины m . Для этого рассмотрим булеву матрицу A размера $m \times l$. Матрица A задает линейное отображение $y \mapsto Ay$ из l -мерного в m -мерное линейного пространства над полем вычетов по модулю 2 (все операции выполняются в поле вычетов по модулю 2).

Лемма 18. Пусть $y_1 \neq y_2$ два различных ненулевых вектора. Тогда случайные величины Ay_1 и Ay_2 независимы и равномерно распределены среди строк длины k (если A выбирается случайно по равномерному распределению).

Доказательство. Равномерная распределенность Ay очевидна. Докажем независимость. Отображение $A \mapsto (Ay_1, Ay_2)$ является линейным отображением из пространства матриц размера $k \times l$ в пространство матриц размера $k \times 2$. Поэтому все элементы в образе этого отображения имеют одинаковую кратность (равную мощности его ядра). Значит, нам достаточно доказать его сюръективность. Нетрудно видеть, что это отображение состоит в умножении справа на матрицу M , состоящую из столбцов

y_1, y_2 . Нам нужно доказать, что ранг этой матрицы равен 2. Это следует из того, что столбцы этой матрицы различны и не равны нулю. \square

Алгоритм декодирования списком основан на той же идее, что и алгоритм исправления менее чем четверти ошибок. Каждый вектор в списке мы будем искать бит за битом. При изложении алгоритма мы опять будем использовать скалярное произведение, введенное в предыдущем параграфе.

Пусть z — код Адамара слова x (составленный из плюс-минус единиц). Пусть \tilde{z} отличается от z инвертированием не более, чем $(1/2 - \varepsilon)2^m$ битов, то есть, $(z, \tilde{z}) \geq 2\varepsilon$.

Нетрудно проверить, что $(z, \tilde{z}^i) = (-1)^{x_i}(z, \tilde{z})$. (Действительно, $(z, \tilde{z}^i) = (z^i, \tilde{z}) = ((-1)^{x_i}z, \tilde{z}) = (-1)^{x_i}(z, \tilde{z})$.) Поэтому, для нахождения x_i достаточно вычислить (z, \tilde{z}^i) с точностью 2ε : если полученное число отрицательно, то $x_i = 1$, а если положительно, то $x_i = 0$.

Как нам найти (z, \tilde{z}^i) с нужной точностью? Проблемы здесь две: во-первых, мы не знаем z , во-вторых, даже при известном z вычисление скалярного произведения требует экспоненциального времени. Вторую проблему решить несложно: ведь нам не нужно точно вычислять (z, \tilde{z}^i) . Можно оценить (z, \tilde{z}^i) вероятностным алгоритмом с помощью закона больших чисел. Для этого рассмотрим величину

$$S_i = \sum_{j=1}^N z(y_j) \tilde{z}^i(y_j),$$

где y_1, \dots, y_N выбираются случайно среди строк длины m . Насколько большим надо взять N , чтобы с большой вероятностью это приближение было 2ε -близко к (z, \tilde{z}^i) ? Для ответа на этот вопрос нам достаточно следующей формы закона больших чисел.

Лемма 19. *Если случайные величины ξ_1, \dots, ξ_N попарно независимы и имеют одинаковое распределение, то для всех положительных δ вероятность события*

$$\left| \frac{\xi_1 + \dots + \xi_N}{N} - \mathbf{E} \xi_i \right| \geq \delta$$

не превосходит $\mathbf{D} \xi_i / (N\delta^2)$. Здесь $\mathbf{E} \xi_i$ и $\mathbf{D} \xi_i$ обозначают математическое ожидание и дисперсию случайной величины ξ_i (по условию они не зависят от i).

Доказательство. Из попарной независимости следует, что дисперсия случайной величины $\xi_1 + \dots + \xi_N$ равна сумме дисперсий ξ_1, \dots, ξ_N , то есть, $N \mathbf{D} \xi_i$. Поскольку случайные величины ξ_1, \dots, ξ_N одинаково распределены, среднее значение суммы $\xi_1 + \dots + \xi_N$ равно $N \mathbf{E} \xi_i$. Поэтому, если указанное в лемме событие имеет место, то величина $(\xi_1 + \dots + \xi_N - N \mathbf{E} \xi_i)^2$ не меньше $\delta^2 N^2$, а значит превосходит свое среднее значение в $N \delta^2 / \mathbf{D} \xi_i$ раз. Вероятность этого не превосходит $\mathbf{D} \xi_i / (N \delta^2)$. \square

Мы будем выбирать y_1, \dots, y_N некоторым хитрым образом. При этом они будут попарно независимы и каждое y_j будет равномерно распределено среди строк длины m . Поэтому среднее значение случайной величины $\xi_j = z(y_j) \tilde{z}^i(y_j)$ равно (z, \tilde{z}^i) . Дисперсия этой величины не превосходит 1 (потому что она принимает значения ± 1). Поэтому вероятность события $|S_i - (z, \tilde{z}^i)| \geq 2\varepsilon$ не превосходит $1/(4N\varepsilon^2)$. Положим $N = \lceil m/2\varepsilon^2 \rceil$. Тогда вероятность этого события будет не больше $1/(2m)$. Значит, с вероятностью не менее $1/2$ для всех $i = 1, \dots, m$ будет выполнено $|S_i - (z, \tilde{z}^i)| < 2\varepsilon$.

Теперь мы укажем способ выбора y_1, \dots, y_N . Положим $l = \lceil \log_2(N + 1) \rceil$ и пусть u_1, \dots, u_N любые различные ненулевые строки из $\{0, 1\}^l$. Возьмем случайную булеву матрицу A размера $m \times l$ и положим $y_j = Au_j$ (операции сложения выполняем по модулю 2). По Лемме 18 вектора y_1, \dots, y_N попарно независимы и равномерно распределены среди строк длины m . А следовательно, по Лемме 19 с вероятностью не менее $1/2$ для всех i выполнено $|S_i - (z, \tilde{z}^i)| < 2\varepsilon$.

Осталось только понять, как можно для данных y_1, \dots, y_N вычислить S_i , не зная z . Здесь нам приходит на помощь особый способ определения y_j . Вспомним, что $y_j = Au_j$, то есть k -ый бит y_j получается умножением k -ой строки A_k матрицы A на вектор u_j , что можно записать, пользуясь нашими обозначениями, как $A_k \odot u_j$. Поэтому

$$z(y_j) = (-1)^{x \odot y_j} = (-1)^{\sum_{k=1}^m x_k (A_k \odot u_j)} = (-1)^{(\sum_{k=1}^m x_k A_k) \odot u_j}.$$

В этом выражении $\sum_{k=1}^m x_k A_k$ есть неизвестный нам вектор длины l . Всего имеется 2^l различных векторов длины l , что меньше $2N$. Поэтому нам не нужно знать z — мы можем перебрать все возможные значения вектора $\sum_{k=1}^m x_k A_k$.

Точнее, пусть w — произвольный вектор длины l . Обозначим через

$S_i(A, w)$ результат подстановки $z(y_j) \rightarrow (-1)^{w \odot u_j}$ и $y_j \rightarrow Au_j$ в сумму

$$S_i = \sum_{j=1}^N z(y_j) \tilde{z}^i(y_j)$$

(сначала делается первая подстановка, а затем — вторая). То есть,

$$S_i(A, w) = \sum_{j=1}^N (-1)^{w \odot u_j} \tilde{z}^i(Au_j).$$

С вероятностью не менее $1/2$ для некоторого w (а именно, для $w = \sum_{k=1}^m x_k A_k$) для всех $i = 1, \dots, m$ будет выполнено $|S_i(A, w) - (z, \tilde{z}^i)| < 2\varepsilon$. Поэтому можно действовать так: выбираем случайно матрицу A ; затем для каждого l -мерного вектора w делаем следующее. Вычисляем суммы $S_i(A, w)$ для всех $i = 1, \dots, m$ и по знаку каждой суммы формируем вектор $x = x(w)$. Затем выдаем список, состоящий из векторов $x(w)$.

3 Шифрование с закрытым ключом

Схема шифрования с закрытым ключом состоит из полиномиально генерируемой последовательности e_n случайных величин и вероятностных полиномиальных алгоритмов E, D , называемых алгоритмами *шифровки* и *расшифровки*. Случайная величина e_n называется *ключом*,

Алгоритмы E и D получают на вход натуральное число n в унарной записи и пару двоичных строк и выдают на выход одну двоичную строку. А именно, алгоритм E получает на вход параметр безопасности n , ключ e_n и сообщение $x \in \{0, 1\}^*$ и выдает некоторую строку $y = E_n(e_n, x)$, называемую *шифrogramмой*, длина которой зависит только от n и длины x . Алгоритм D получает на вход n, y и e_n и выдает строку $D_n(e_n, y)$.

Мы хотим, чтобы при любом x $D(e_n, E_n(e_n, x)) = x$ с примерно единичной вероятностью, причем при случайно выбранном ключе e_n шифrogramма $E_n(e_n, x)$ не должна нести никакой информации об x . Эти требования уточняются следующим образом.

(1) Правильность расшифровки: для любой последовательности сообщений x_n длины $\text{poly}(n)$ с вероятностью приблизительно равной 1 выполнено $D(e, E(e, x)) = x$. (Для упрощения обозначений мы опускаем индекс n .) Источником случайности здесь является не только случайная

величина e , но и случайные биты алгоритмов E, D . Мы предполагаем, что случайные биты алгоритмов E, D независимы от e .

(2) Неразглашение информации: для любого полинома $p(n)$ и для любых двух последовательностей сообщений a_n, b_n таких, что длины a_n, b_n равны $p(n)$, случайные величины $E(e, a_n)$ и $E(e, b_n)$ вычислительно неотличимы. Источник случайности в этом определении имеется в e и в случайных битах алгоритмов E, D , которые предполагаются независимыми от e .

Условие (2) очевидно эквивалентно своему частному случаю: неотличимости случайных величин $E(e, a_n)$ и $E(e, 00 \dots 0)$ (подряд $p(n)$ нулей). Вторую из этих случайных величин можно генерировать, зная только n . Поэтому, какое бы сообщение мы не зашифровывали случайно выбранным ключом, шифрограмма не будет нести никакой информации — некоторую неотличимую от нее случайную величину можно сгенерировать и не зная исходного сообщения. Заметим, что в наших определениях важно, что ключ выбирается случайно, а не фиксирован. Мы не можем требовать неотличимости шифрограмм $E(e, a_n)$ и $E(e, b_n)$ при любом фиксированном ключе e , поскольку любой фиксированный ключ e можно “запаять” в схему-отличитель, которая будет применять алгоритм D и, скажем, выдавать первый бит расшифрованного сообщения.

Из условия (2) следует, что, не зная ключа, никакой бит исходного сообщения нельзя найти по шифрограмме с вероятностью существенно больше $1/2$, если сообщение выбирается случайно с равномерным распределением среди всех строк какой-то фиксированной полиномиальной от n длины k . Действительно, пусть противник C (схема полиномиальной от n размера) пытается по шифрограмме восстановить, скажем, первый бит сообщения. Из условия (2) следует, что вероятности событий $C(E(e, 0a)) = 1$ и $C(E(e, 1a)) = 1$ (где a случайная строка длины $k - 1$) приблизительно равны. Другими словами сумма вероятностей событий $C(E(e, 0a)) = 0$ и $C(E(e, 1a)) = 1$ приблизительно равна 1, следовательно, вероятность события $C(E(e, b)) = b[1]$ (где b случайная строка длины k) примерно равна $1/2$.

Задача 29. Пусть противник интересуется некоторой информацией $f(x)$ о передаваемом сообщении x (где $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$ — некоторая функция). Допустим противник применяет к шифрограмме $E(e, x)$ некоторую схему D_n полиномиальной от n размера для извлечения этой информации. Докажите, что существует доступная случайная величина α_n та-

кая, что для любой последовательности сообщений x_n полиномиальной длины, вероятности событий $D_n(E(e, x_n)) = f(x_n)$ (вероятность успеха атаки) и $\alpha_n = f(x_n)$ приблизительно равны. То есть, информация $f(x_n)$ может быть вычислена и без подслушивания с примерно той же вероятностью успеха.

Пара алгоритмов E, D и последовательность случайных величин e_n называется *схемой шифрования с закрытым ключом*, если выполнены все перечисленные условия. Схему шифрования с закрытым ключом можно построить на основе любого генератора ПСЧ.

Теорема 20. *Если существует генератора G_n , то существует и схема шифрования с закрытым ключом.*

Доказательство. Сначала построим схему шифрования с закрытым ключом, для которой свойство (2) выполнено только для какого-нибудь одного фиксированного полинома $p(n)$ (а не для всех полиномов, как требуется). Эта схема называется *гаммированием (one-time pad)*. В качестве ключа возьмем случайную строчку γ_n длины $p(n)$ и положим $E(\gamma_n, x) = \gamma_n \oplus x$ (побитовое сложение по модулю 2) и $D(\gamma_n, y) = \gamma_n \oplus y$. (Если длина сообщения не равна $p(n)$, то положим $E(\gamma_n, x) = x$, $D(\gamma_n, y) = y$; второе условие в этом случае, разумеется, не выполнено, но это и не требуется.) Условие (1) очевидно выполнено. В условии (2) вероятность события $C_n(\gamma_n \oplus a_n) = 1$ равна вероятности события $C_n(\gamma_n) = 1$, а потому не зависит от a_n . При этом нам даже не важно, что схема C_n имеет полиномиальный размер — случайные величины $\gamma_n \oplus a_n$ и γ_n имеют одинаковое распределение.

Недостатком гаммирования является то, что длина ключа равна длине сообщения, поэтому на практике его можно применять только для шифрования коротких сообщений.

Теперь модифицируем схему гаммирования, заменив случайную величину γ_n на псевдослучайную величину, порожденную генератором. Ключ s равномерно распределен среди строк длины n . Алгоритм $E(s, x)$ является детерминированным и выдает побитовую сумму x и первых $|x|$ битов последовательности $G_n(s)$. Будем обозначать выданную им последовательность через $x \oplus G(s)$. Алгоритм $D(s, y)$ выдает $y \oplus G(s)$, побитовую сумму y и первых $|y|$ битов последовательности $G(s)$. Ясно, что условие (1) выполнено. Для проверки условия (2) фиксируем произвольный полином $p(n)$. Нам надо доказать неотличимость случайных величин $a_n \oplus G(s)$ и $b_n \oplus G(s)$, где a_n, b_n любые последовательности слов длины

$p(n)$. Поскольку функция G является генератором, случайная величина $G(s)[1 : p(n)]$ вычислительно неотличима от случайной величины $\gamma_{p(n)}$, равномерно распределенной среди слов длины $p(n)$. Значит, случайная величина $a_n \oplus G(s)$ неотличима от случайной величины $a_n \oplus \gamma_{p(n)}$. В самом деле, если бы они были отличимы тестом $T_n(y)$ размера $\text{poly}(n)$ то тест $T_n(a_n \oplus y)$, размер которого лишь немного превышает размер T_n , отличал бы $G(s)$ и γ . Случайная величина $a_n \oplus \gamma_{p(n)}$ имеет равномерное распределение, значит случайная величина $a_n \oplus G(s)$ неотличима от случайной величины $\gamma_{p(n)}$. То же самое справедливо для случайной величины $b_n \oplus G(s)$, значит она неотличима от $a_n \oplus G(s)$. \square

4 Псевдослучайные функции

Пусть задано семейство функций $f_s^n : \{0, 1\}^{k(n)} \rightarrow \{0, 1\}^{l(n)}$, где $s \in \{0, 1\}^{m(n)}$, а $m(n), k(n), l(n)$ — некоторые многочлены. Будем называть его семейством псевдослучайных функций (ПСФ), если

(1) отображение $(1^n, s, x) \mapsto f_s^n(x)$ вычислимо за полиномиальное время и

(2) для любого полинома $p(n)$ и любой последовательности из $p(n)$ различных слов $x_1^n, \dots, x_{p(n)}^n$ длины $k(n)$ случайная величина $f_s^n(x_1^n) \dots f_s^n(x_{p(n)}^n)$ (где s выбирается равномерно среди всех строк длины $m(n)$) вычислительно неотличима от случайной величины $r_1 \dots r_{p(n)}$, равномерно распределенной среди всех строк длины $p(n)l(n)$.

Теорема 21. *Если существует генератор ПСЧ $G_n : \{0, 1\}^n \rightarrow \{0, 1\}^{n+1}$, то для любых полиномов $k(n), l(n)$ существует семейство ПСФ $f_s^n : \{0, 1\}^{k(n)} \rightarrow \{0, 1\}^{l(n)}$.*

Доказательство. Напомним, что если существует генератор ПСЧ $n \rightarrow n+1$, то для любых полиномов p, q существует генератор ПСЧ $p(n) \rightarrow q(n)$. Поэтому существует генератор $G_n : \{0, 1\}^{l(n)} \rightarrow \{0, 1\}^{2l(n)}$.

Положим $m(n) = l(n)$. Пусть s произвольная строка длины $l(n)$ и $x = x_1 \dots x_k$ произвольное слово длины $k(n)$. Значение $f_s^n(x)$ определяется так. Разрежем слово $G(s)$ на две половины, и обозначим их через s_0 и s_1 . Рассмотрим первую половину, если $x_1 = 0$ и вторую половину, если $x_1 = 1$. Опять применим к этому слову (s_{x_1}) отображение G и разрежем полученное слово на две части. Обозначим их через $s_{x_1 0}, s_{x_1 1}$. Теперь

опять рассмотрим слово $s_{x_1x_2}$ и так далее. На каждом шаге мы применяем к текущему слову s_w отображение G и оставляем левую половину, если очередной бит слова x равен нулю, и вторую половину, иначе.

В результате мы получим некоторое слово s_x , которое и будет значением f_s на x . Очевидно, что $f_s(x)$ можно найти за полиномиальное время. Проверим условие (2).

Пусть $x_1, \dots, x_{p(n)}$ последовательность из $p(n)$ различных слов длины $k(n)$. Нам нужно доказать вычислительную неотличимость $f_s(x_1) \dots f_s(x_{p(n)})$ и $r_1 \dots r_{p(n)}$.

Для этого рассмотрим новое семейство функций $\tilde{f}_{t_0t_1}$, каждая функция в котором задается строкой t_0t_1 длины $2l$ (а не длины l , как раньше). Новое семейство будем определять в точности, как и раньше, за исключением первого шага в рекуррентном определении s_w (для слов w длины не более k). А именно, слова s_0, s_1 теперь положим равными t_0, t_1 , соответственно. Слово $s_{\text{пустое слово}}$ теперь не определено. Мы утверждаем, что случайные величины $f_s(x_1) \dots f_s(x_{p(n)})$ и $\tilde{f}_{t_0t_1}(x_1) \dots \tilde{f}_{t_0t_1}(x_{p(n)})$ вычислительно неотличимы. Действительно, если бы они были отличимы схемой C , то эту схему можно было бы использовать для отличения случайных величин $G(s) = s_0s_1$ и t_0t_1 .

Теперь тот же трюк можно проделать еще раз: в определении $\tilde{f}_{t_0t_1}$ можно заменить на s_{00}, s_{01} на независимые равномерно распределенные слова длины l (слово s_0 после этой замены становится неопределенным). Затем можно будет, скажем, заменить s_{010}, s_{011} на независимые равномерно распределенные слова и так не более, чем полином раз. На каждом шаге мы можем сделать следующее: если в определении текущего семейства \tilde{f}_t слово s_w выбирается случайным образом (как подслово строки t , задающей функцию), мы можем заменить способ определения s_{w0}, s_{w1} : вместо того, чтобы применять к s_w генератор G , мы можем взять случайные и независимые строки. При этом количество битов в t увеличится на l . При каждом изменении семейства \tilde{f}_t случайная величина $\tilde{f}_t(x_1) \dots \tilde{f}_t(x_{p(n)})$ будет изменяться незначительно: измененная случайная величина будет вычислительно неотличима от исходной.

Если делать это не более чем полином раз, то для результирующего семейства \tilde{f}_t случайная величина $\tilde{f}_t(x_1) \dots \tilde{f}_t(x_{p(n)})$ будет вычислительно неотличима от исходной случайной величины $f_s(x_1) \dots f_s(x_{p(n)})$. Применим это преобразование для всех начал w слова x_1 (в порядке возрастания длины w), затем для всех начал слова x_2 и так далее. Всего мы

произведем не более $k(n)p(n)$ изменений. Полученная случайная величина $\tilde{f}_t(x_1) \dots \tilde{f}_t(x_{p(n)})$ вычислительно неотличима от случайной величины $f_s(x_1) \dots f_s(x_{p(n)})$. Осталось заметить, что $\tilde{f}_t(x_1) \dots \tilde{f}_t(x_{p(n)})$ имеет равномерное распределение, поскольку $\tilde{f}_t(x_i)$ по построению есть подслово t и при разных i эти подслова не перекрываются. \square

Построенное в доказательстве семейство функций удовлетворяет следующему более сильному требованию.

(2') Пусть дана последовательность схем C_n полиномиального размера. Схема сначала находит слово x_1 , затем, получая $f(x_1)$, находит x_2 и так далее — всего $p(n)$ раз (где p — некоторый полином). Точнее $x_1 = C_n$ (пустое слово), $x_2 = C_n(f(x_1))$, \dots , $x_{p(n)} = C_n(f(x_1) \dots f(x_{p(n)-1}))$. Положим $C^f = C(f(x_1) \dots f(x_{p(n)}))$. Тогда вероятность события $C^{f_s} = 1$ приблизительно равна вероятности события $C^g = 1$, где g — случайная функция из $\{0, 1\}^{k(n)}$ в $\{0, 1\}^{l(n)}$ (все функции g считаются равновероятными).

Теорема 22. *Построенное в доказательстве предыдущей теоремы семейство ПСФ $f_s^n : \{0, 1\}^{k(n)} \rightarrow \{0, 1\}^{l(n)}$ удовлетворяет требованию (2').*

Доказательство. Пусть дана схема C и полином $p(n)$. Будем определять промежуточные семейства \tilde{f}_t примерно так же, как в предыдущем доказательстве. А именно, найдем $x_1 = C_n$ (пустое слово). Затем применим преобразования из предыдущего доказательства для всех собственных начал w слова x_1 . В результирующем семействе \tilde{f}_t значение $\tilde{f}_t(x_1)$ выбирается случайным образом. Слово x_2 определим как $C(\tilde{f}_t(x_1))$. Оно уже зависит не только от схемы C , но и от t . И опять применим преобразование из предыдущего доказательства для всех собственных начал w слова x_2 , которые не являются началом x_1 . После этого определим x_3 и так далее. При каждом изменении семейства \tilde{f}_t вероятность события $C^{\tilde{f}_t} = 1$ меняется незначительно. А для заключительного семейства \tilde{f}_t вероятность события $C^{\tilde{f}_t} = 1$ совпадает с вероятностью события $C^g = 1$ для случайно выбранной функции g . Действительно, если очередное x_i не совпадает ни с одним из x_1, \dots, x_{i-1} , то в качестве $\tilde{f}_t(x_i)$ выбирается случайная строка, не зависящая от предыдущих. \square

Аналогичным образом можно построить семейство ПСФ $f_s^n : \{0, 1\}^* \rightarrow \{0, 1\}^{l(n)}$ удовлетворяющих требованиям (1) и (2'). Для этого сопоставим

каждому слову $x = x_1x_2 \dots x_m$ его префиксный код $\hat{x} = x_1x_1x_2x_2 \dots x_mx_m01$. Этот код обладает следующим свойством: если $x \neq z$, то ни одно из слов \hat{x}, \hat{z} не является началом другого. Положим $m(n) = l(n)$ и $f_s^n(x) = s_{\hat{x}}$ (отображение $(w, s) \mapsto s_w$ определено в доказательстве Теоремы 21). Поясним, почему мы в определении заменили x на его префиксный код. Это сделано потому, что для любых слов u, v слово s_{uv} является легко вычислимой функцией слова s_u , и поэтому старое определение не годится.

Теорема 23. *Построенное семейство $f_s^n : \{0, 1\}^* \rightarrow \{0, 1\}^{l(n)}$ удовлетворяет условиям (1) и (2').*

Доказательство. Условие (1) очевидно. Проверим (2'). Пусть дана схема C и полином $p(n)$. Будем определять промежуточные семейства \tilde{f}_t так же, как в предыдущем доказательстве. А именно, найдем $x_1 = C_n$ (пустое слово). Затем применим преобразования из доказательства Теоремы 21 для всех собственных начал w слова \hat{x}_1 . В результирующем семействе \tilde{f}_t значение $\tilde{f}_t(x_1)$ выбирается случайным образом. Слово x_2 определим как $C(\tilde{f}_t(x_1))$. Если $x_2 = x_1$, то ничего не делаем. Иначе применим преобразование из доказательства Теоремы 21 для всех собственных начал w слова \hat{x}_2 , которые не являются началом x_1 . По свойству префиксности отображения $x \mapsto \hat{x}$ для полученного семейства \tilde{f}_t значение $\tilde{f}_t(x_2) = \tilde{s}_{\hat{x}_2}$ определено и выбирается случайным образом, независимо от $\tilde{f}_t(x_1)$. После этого определим x_3 и так далее. При каждом изменении семейства \tilde{f}_t вероятность события $C^{\tilde{f}_t} = 1$ меняется незначительно. А для заключительного семейства \tilde{f}_t вероятность события $C^{\tilde{f}_t} = 1$ совпадает с вероятностью события $C^g = 1$ для случайно выбранной функции g . Действительно, если очередное x_i не совпадает ни с одним из x_1, \dots, x_{i-1} , то в качестве $\tilde{f}_t(x_i)$ выбирается случайная строка, не зависящая от предыдущих. \square

5 Шифрование с закрытым ключом многих сообщений

6 Односторонние перестановки с секретом и шифрование с открытым ключом

Определение схемы шифрования с открытым ключом

В схеме шифрования с открытым ключом имеется два ключа — открытый и закрытый. Открытый ключ используется для шифровки, а закрытый для расшифровки. От схемы шифрования требуется, чтобы противник из открытого ключа и шифрограммы любого сообщения не смог получить никакой информации о сообщении. Точнее, для любых двух сообщений x_1, x_2 одной длины случайная величина (e, c_1) , состоящая из открытого ключа e и шифрограммы s_1 сообщения x_1 , была вычислительно неотличима от случайной величины (e, c_2) , состоящей из открытого ключа и шифрограммы сообщения x_2 . Открытый ключ можно опубликовать, тогда любой может посылать зашифрованные сообщения, расшифровывать которые может только владелец закрытого ключа.

Схема шифрования с открытым ключом состоит из полиномиальных вероятностных алгоритмов E и D , называемых, алгоритмами *шифровки* и *расшифровки* и полиномиально генерируемой последовательности случайных величин $\langle e_n, d_n \rangle$. Строки e_n и d_n называются *открытым и закрытым ключами*.

Алгоритм E получает на вход 1^n , ключ e_n и сообщение $x \in \{0, 1\}^*$ и выдает некоторую строку $y = E_n(e_n, x)$, называемую *шифrogramмой*, длина которой зависит только от длин n и x . Алгоритм D получает на вход 1^n , y и d и выдает некоторую строку $D_n(d_n, y)$. Тройка $(E, D, \langle e_n, d_n \rangle)$ должна удовлетворять следующим требованиям.

(1) Правильность расшифровки: для любой последовательности сообщений x_n длины $\text{poly}(n)$ с вероятностью приблизительно равной 1 выполнено $D(e, E(d, x)) = x$ (то есть, вероятность отличается от 1 на пренебрежимо малую величину).⁶ Источником случайности здесь является не только случайная величина $\langle e, d \rangle$, но и случайные биты алгоритмов E, D , которые предполагаются независимыми в совокупности.

⁶ Для упрощения обозначений мы опускаем индекс n .

(2) Для любого полинома $p(n)$ и для любых двух последовательностей сообщений a_n, b_n таких, что длины a_n, b_n равны $p(n)$, случайные величины $\langle e, E(e, a_n) \rangle$ и $\langle e, E(e, b_n) \rangle$ вычислительно неотличимы. Источником случайности здесь является не только случайная величина $\langle e, d \rangle$, но и случайные биты алгоритма E . Мы предполагаем, что случайные биты алгоритма E независимы от $\langle e, d \rangle$.

Пара алгоритмов E, D и последовательность случайных величин $\langle e_n, d_n \rangle$ называется *схемой шифрования с открытым ключом*, если выполнены все перечисленные условия. Для существования схемы шифрования с открытым ключом требуется более сильная гипотеза, чем существование генератора. Такую схему удастся построить при условии существования односторонней перестановки с секретом (определение дано в следующем разделе).

Односторонние перестановки с секретом

На самом деле речь идет не об одной функции, а о семействе функций. Грубо говоря, необратимой перестановкой с секретом (trapdoor permutation) называется семейство перестановок $f_n^e : D_n^e \rightarrow D_n^e$ такие, что функция $\langle e, x \rangle \mapsto \langle e, f_n^e(x) \rangle$ односторонняя, но при этом существует полиномиальный алгоритм, который по $f_n^e(x)$ и некоторой дополнительной информации d , зависящей только от e , находит x .

Теперь формальное определение. Пусть задана полиномиально генерируемая последовательность случайных величин $\langle e_n, d_n \rangle$. Будем через A_n обозначать первые компоненты всех возможных значений $\langle e_n, d_n \rangle$. Пусть для каждого $e \in A_n$ задана перестановка f_n^e некоторого множества слов D_n^e длины $\text{poly}(n)$. Последовательность пар $\langle \langle e_n, d_n \rangle, \{f_n^e\} \rangle$ называется односторонней перестановкой с секретом, если выполнены следующие условия.

(1) Функция $\langle 1^n, e, x \rangle \mapsto f_n^e(x)$ вычислима за полиномиальное от n время (существует полиномиальный алгоритм, который по любой тройке $\langle 1^n, e, x \rangle$, где $e \in A_n, x \in D_n^e$, вычисляет $f_n^e(x)$).

(2) Для любой последовательности схем C_n размера $\text{poly}(n)$ вероятность того, что C_n по $\langle e, f_e(x) \rangle$ найдет x стремится к нулю быстрее любого обратного полинома от n . Здесь e выбирается случайно, как первая компонента случайной величины $\langle e_n, d_n \rangle$, а x выбирается независимо от e из D^e (все слова из D^e предполагаются равновероятными).

(3) Обозначим через γ_n^e случайную величину, равномерно распределенную в D_n^e . Существует вероятностный полиномиальный алгоритм, который по 1^n и любому $e \in A_n$ генерирует случайную величину ξ_n^e такую, что случайная величина $\langle e_n, \xi_n^e \rangle$ статистически неотличима от случайной величины $\langle e_n, \gamma_n^e \rangle$.

Из этого условия следует доступность случайной величины $\langle e_n, \gamma_n^e \rangle$ (сначала генерируем e_n , а затем γ_n^e , пользуясь условием (3)). Поэтому из условий (1), (2), (3) следует, что функция $\langle e, x \rangle \mapsto \langle e, f_n^e(x) \rangle$ является обобщенной односторонней перестановкой.

(4) Существует полиномиальный вероятностный алгоритм, который по тройке $\langle 1^n, d, f_e(x) \rangle$ с вероятностью приблизительно равной 1 вычисляет x . Здесь пара $\langle e, d \rangle$ выбирается генератором случайной величины $\langle e_n, d_n \rangle$, а x выбирается равномерно в D_n^e .

Поскольку неизвестно, существуют ли односторонние перестановки, тем более неизвестно, существуют ли односторонние перестановки с секретом. Гипотеза об их существовании довольно правдоподобна. Действительно, если функции Рабина или функция RSA в самом деле необратима, то существуют односторонние функции с секретом. Убедимся в этом.

Функция Рабина. Случайная величина $\langle e_n, d_n \rangle$ равномерно распределена в множестве всех пар $\langle pq, \langle p, q \rangle \rangle$, где p, q — n -битовые простые числа вида $4k + 3$ (точнее, полиномиально генерируемая величина, статистически неотличимая от этой). Множество D_n^e есть множество всех квадратичных вычетов по модулю pq , принадлежащих интервалу $0, \dots, pq - 1$, при этом $f_n^e(x) = x^2 \pmod{pq}$. Условия (1), (3) и (4) очевидно выполнены. Условие (2) будет выполнено, если функция Рабина необратима.

Функция RSA. Случайная величина $\langle e_n, d_n \rangle$ равномерно распределена в множестве всех пар $\langle \langle pq, i \rangle, \langle pq, j \rangle \rangle$, где p, q — n -битовые простые числа, $i, j \in \{1, \dots, pq - 1\}$ и $ij \equiv 1 \pmod{(p-1)(q-1)}$. (Точнее, полиномиально генерируемая величина, статистически неотличимая от этой.) Множество D_n^e есть множество всех $0 < x < pq$, взаимно простых с pq , при этом $f_n^e(x) = x^i \pmod{pq}$. Условия (1), (3) и (4) очевидно выполнены. Условие (2) будет выполнено, если функции RSA в самом деле необратима.

Построение схемы шифрования с открытым ключом

Теорема 24. *Если существует односторонняя перестановка с секретом, то существует и схема шифрования с открытым ключом.*

Доказательство. Сначала научимся шифровать один бит. Гаммирование нам уже не поможет, потому что в схеме гаммирования закрытый ключ совпадал с открытым. Нам опять понадобится понятие трудного бита. Пусть имеется необратимая перестановка с секретом. Назовем семейство функций $h_n^e : D_n^e \rightarrow \{0, 1\}$, трудным битом к f_n^e , если случайная величина $h_n^{e_n}(x)$ трудно вычислима по случайной величине $\langle f_n^{e_n}(x), e_n \rangle$ (где x выбирается равномерно из $D_n^{e_n}$ независимо от e_n).

Лемма 25. *Если существует односторонняя перестановка с секретом, то существует и односторонняя перестановка и секретом и трудный бит для нее.*

Доказательство. Это следует из теоремы Левина–Голдрайха. Пусть f^e исходная односторонняя перестановка с секретом. В новой односторонней перестановке случайная величина $\langle e_n, d_n \rangle$ та же самая, а функция g^e определяется как $g^e(xy) = f^e(x)y$, где y строка из нулей и единиц той же длины, что и x (мы опускаем параметр безопасности n в обозначениях). Таким образом область определения g^e состоит из всех слов вида xy , где $x \in D^e$, $|y| = |x|$. Трудный бит h^e определяется как скалярное произведение: $h^e(xy) = x \odot y$.

Условия (1)–(4) очевидно выполнены для g^e . Докажем, что h^e трудный бит к g^e . Предположим, что схема C_n полиномиального от n размера с вероятностью $1/2 + \varepsilon_n$ по $e, f^e(x), y$ вычисляет $x \odot y$. Тогда ее можно переделать в схему полиномиального размера, которая по $e, f^e(x), y, z$ (где z равномерно распределено среди слов той же длины, что e , и независимо от x, y, e) вычисляет с вероятностью $1/2 + \varepsilon_n$ скалярное произведение конкатенации ex и zy , поскольку последнее равно $e \odot z \oplus z \odot y$. Из условия (2) следует, что указанное распределение на тройках $\langle e, x, y \rangle$ является трудным для функции $(e, x, y, z) \mapsto (e, g^e(x), y, z)$. По теореме 16 ε_n пренебрежимо мало. \square

Теперь, имея необратимую функцию g^e с секретом и трудный бит h^e для нее, построим схему шифрования с открытым ключом одного бита b . Случайную величину $\langle e_n, d_n \rangle$ заимствуем из необратимой функции.

Алгоритм шифрования выдает пару $y = \langle b \oplus h^e(x), g^e(x) \rangle$. Здесь x выбирается по распределению ξ^e из условия (3) для функции g^e .

Зная d расшифровать y просто: надо найти x , обратив $g^e(x)$, и сложить первую компоненту y с $h^e(x)$. Вероятность ошибки при расшифровке складывается из статистического расстояния между случайными величинами (e, ξ^e) и (e, γ^e) (где γ^e равномерно распределено в D^e) и вероятности ошибки алгоритма из условия (4). Обе вероятности по условию пренебрежимо малы.

Шифрограмма любого из двух битов вместе с открытым ключом e будет вычислительно неотличима от случайной величины $\langle r, g^e(x), e \rangle$, где r случайный равномерно распределенный бит, независимый от x, e . Действительно, по лемме 7 случайная величина $\langle r, g^e(x), e \rangle$ вычислительно неотличима от случайной величины $\langle h^e(x), g^e(x), e \rangle$. Поэтому для любого бита b случайные величины $\langle b \oplus r, g^e(x), e \rangle$ и $\langle b \oplus h^e(x), g^e(x), e \rangle$ вычислительно неотличимы. Первая из них имеет то же распределение, что и $\langle r, g^e(x), e \rangle$. При замене равномерного распределения на D^e на распределение, генерируемое алгоритмом из пункта (3), получим случайную величину, статистически неотличимую от исходной.

Теперь построим схему шифрования произвольного числа битов. Опять заимствуем алгоритм генерации ключей из необратимой функции. Шифрограмму сообщения $b_1 b_2 \dots b_m$ из m битов определим как

$$b_1 \oplus h^e(x), b_2 \oplus h^e(g^e(x)), \dots, b_m \oplus h^e((g^e)^{m-1}(x)), (g^e)^m(x).$$

Все условия, кроме условия (2), очевидны. Условие (2) следует из леммы 10. Пусть $p(n)$ произвольный полином. Применяя лемму 10 к функциям $\langle e, x \rangle \mapsto \langle e, g^e(x) \rangle$, $\langle e, x \rangle \mapsto h^e(x)$ и случайной величине $\alpha_n = \langle e, \gamma^e \rangle$, мы можем заключить, что случайные величины

$$e, h^e(\gamma^e), h^e(g^e(\gamma^e)), \dots, h^e((g^e)^{p(n)-1}(\gamma^e)), (g^e)^{p(n)}(\gamma^e)$$

и $e, r_1, r_2, \dots, r_{p(n)}, \gamma^e$ вычислительно неотличимы.

Отсюда следует вычислительная неотличимость случайных величина

$$e, b_1 \oplus h^e(\gamma^e), b_2 \oplus h^e(g^e(\gamma^e)), \dots, b_{p(n)} \oplus h^e((g^e)^{p(n)-1}(\gamma^e)), (g^e)^{p(n)}(\gamma^e)$$

и $e, b_1 \oplus r_1, b_2 \oplus r_2, \dots, b_{p(n)} \oplus r_{p(n)}, \gamma^e$ вычислительно неотличимы. Последняя из них имеет то же распределение, что и случайная величина

$$e, r_1, r_2, \dots, r_{p(n)}, \gamma^e,$$

которая не зависит от $b_1 \dots, b_{p(n)}$. Поэтому при разных сообщениях $b_1 \dots, b_{p(n)}$ случайные величины, состоящие из открытого ключа и шифрограммы, вычислительно неотличимы друг от друга. \square

Задача 30. Докажите, что можно из любой схемы шифрования с открытым ключом одного бита можно изготовить схему шифрования с открытым ключом произвольного количества битов с помощью шифрования каждого бита по отдельности с помощью одного и того же открытого ключа. Объясните, почему для схем шифрования с закрытым ключом, то же самое неверно.

7 Протоколы привязки к биту (bit commitment)

Протоколы привязки к биту имитируют сундучок с замком, в который первый игрок (отправитель) может положить один бит, закрыть его на ключ и передать второму игроку (получателю). Получатель не может открыть сундучок до тех пор, пока отправитель не передаст ему ключ. Отправитель не может подменить бит, положенный в сундучок. Различают два вида таких протоколов — интерактивные и неинтерактивные. Начнем со вторых, как более простых.

Неинтерактивный протокол привязки к биту состоит из двух алгоритмов: вероятностного полиномиального алгоритма S и детерминированного полиномиального алгоритма R . Алгоритм S получает на вход параметр безопасности n и один бит σ и выдает на выход ключ $k = k_n(\sigma, r)$ и привязку $c = c_n(\sigma, r)$ (r — использованные алгоритмом S случайные биты); привязка имитирует закрытый сундучок. Алгоритм R получает на вход ключ k и привязку c и выдает на выход один бит $R_n(c, k)$ или специальный символ \perp (последнее означает, что ключ k не подошел к замку).

Требования к этим алгоритмам такие.

(1) Для любого n и любого бита σ с вероятностью, приблизительно равной 1, выполнено $R_n(c_n(\sigma, r), k_n(\sigma, r)) = \sigma$ (настоящий ключ открывает сундучок).

(2) Для любого n не существует таких c, k', k'' , что $R_n(c, k') = 0$ и $R_n(c, k'') = 1$ (любая строка c может быть привязкой только к одному биту).

(3) Последовательности случайных величин $c_n(0, r)$ и $c_n(1, r)$ вычислительно неотличимы (не зная ключа, получатель не имеет никакой информации о содержимом сундучка).

Если выполнены все три условия, то пара алгоритмов R, S называется протоколом привязки к биту.

Теорема 26. *Если существует односторонняя инъективная частичная функция $f_n : D_n \rightarrow \{0, 1\}^*$, область определения которой разрешима за время $\text{poly}(n)$ (то есть, существует алгоритм, который по любой паре $\langle n, x \rangle$ за время $\text{poly}(n)$ определяет, принадлежит ли x к D_n), то существует и протокол привязки к биту.*

Доказательство. По теореме Левина-Голдрейха из условия теоремы следует существование функции g , удовлетворяющей условию теоремы и трудного бита h к ней. Алгоритм S выбирает случайную строку x из области определения g_n по полиномиально генерируемому распределению, статистически неотличимому от равномерного распределения на D_n . Ключ k равен x , а привязка s равна $\langle \sigma \oplus h(x), g(x) \rangle$ (мы опускаем индекс n). Алгоритм R , получив на вход x и $\langle \delta, y \rangle$, проверяет равенство $g(x) = y$ и то, что $x \in D_n$. Если равенство неверно или $x \notin D_n$, он выдает \perp . Иначе он выдает $\delta \oplus h(x)$.

Условие (1) очевидно выполнено. Условие (2) следует из инъективности g . Действительно, если $g(x') = y = g(x'')$, и $x', x'' \in D_n$, то $x' = x''$, следовательно, $\delta \oplus h(x') = \delta \oplus h(x'')$.

Условие (3) означает вычислительную неотличимость случайных величин $\langle h(x), g(x) \rangle$ и $\langle 1 \oplus h(x), g(x) \rangle$ (где x выбирается равномерно из D_n). Это следует из того, что они обе неотличимы от случайной величины $\langle \gamma, g(x) \rangle$, где γ случайный бит, независимый от x . \square

Задача 31. Докажите, что без ограничения общности можно считать, что в неинтерактивных протоколах привязки к биту ключ всегда равен σr (где σ — исходный бит, а r случайные биты алгоритма S).

Задача 32. Убедитесь, что теорема остается верной и для обобщенных односторонних инъективных функций с полиномиально разрешимой областью определения.

К сожалению, что область определения всех кандидатов в односторонние инъективные функции не является полиномиально разрешимой (точнее, неизвестен полиномиальный алгоритм, ее разрешающий). Поэтому, неясно, к какой функции можно было бы применить теорему 26.

Перейдем поэтому к интерактивным протоколам. В интерактивном протоколе получатель также участвует в положении бита в сундучок. Отправитель и получатель обмениваются сообщениями, которые записываются на магнитофонную ленту. Эта запись и играет роль привязки к биту. Определение интерактивного протокола привязки к биту сложнее, чем неинтерактивного. Но зато интерактивный протокол можно построить, исходя из любого генератора псевдослучайных чисел.

7.1 Интерактивные алгоритмы

Интерактивными алгоритмы — это алгоритмы, которые, кроме обычного входа, имеют еще входной и выходной каналы обмена информацией. Чтобы дать определение интерактивному алгоритму, начнем с понятия стратегии. Стратегией называется функция, отображающая конечные последовательности слов в бинарном алфавите в слова в бинарном алфавите. Пусть имеются стратегии F, G . Последовательность сообщений c_1, c_2, \dots между стратегиями определяется по правилу

$$\begin{aligned} c_1 &= F(), \\ c_2 &= G(c_1), \\ c_3 &= F(c_1, c_2), \\ c_4 &= G(c_1, c_2, c_3), \\ &\dots \end{aligned}$$

Общение длится до тех пор, пока некоторое c_i не станет пустым; посылка пустого слова означает желание прекратить общение. Если пустое слово не послано, то последовательность сообщений бесконечна. Последовательность сообщений c_1, c_2, c_3, \dots будем обозначать через $c(F, G)$ и называть диалогом между F и G .

Стратегией с параметром из некоторого множества M называется отображение, сопоставляющее каждому элементу $x \in M$ некоторую стратегию F_x . Интерактивным алгоритмом со входами из $\{0, 1\}^*$ называется вычислимая стратегия с параметром из $\{0, 1\}^*$. Чтобы запустить интерактивный алгоритм A , задаваемый стратегией с параметром F_x , нужен вход и собеседник, то есть, некоторая стратегия. Пусть x бинарное слово, а G — некоторая стратегия. Результат $A^G(x)$ работы алгоритма A на входе x с собеседником G определяется следующим образом. Запустим

стратегии F_x и G . Пусть $c(F_x, G)$ полученный диалог. Если диалог бесконечен, то результат не определен. Иначе применим к $c(F_x, G)$ стратегию F_x , полученное слово и будет результатом. Пусть A, B два интерактивных алгоритма, а x, y двоичные слова. Запустим алгоритм A на входе x , а B на входе y так, чтобы они общались друг с другом. Результат, выданный алгоритмом A будем обозначать через $A^{B(y)}(x)$.

Обозначение $A^G(x)$ напоминает обозначения, связанные с машинами с оракулом. Подчеркнем важное отличие машин с оракулом от интерактивных алгоритмов. Ответ оракула на очередной вопрос зависит только от этого вопроса. Очередное сообщение стратегии зависит от всех предыдущих сообщений полученных от алгоритма.

Теперь определим понятие полиномиального интерактивного алгоритма. Полиномиальной стратегией F_x с параметром из некоторого множества двоичных слов называется стратегия, вычисляемая за полиномиальное от $n = |x|$ время. Такая стратегия может сделать только полиномиальное от n количество ходов, полиномиальной от n длины (точнее, сделанные ей ходы могут зависеть только от полинома первых символов последовательности сделанных ходов). Интерактивный алгоритм A называется полиномиальным, если стратегия с параметром, его задающая, полиномиальна.

Полиномиальные вероятностные интерактивные алгоритмы (и стратегии с параметром) можно определить через полиномиальные детерминированные интерактивные алгоритмы. Пусть A полиномиальный детерминированный интерактивный алгоритм, а p некоторый полином. Будем понимать результат работы A на входе вида xr с собеседником G , где r двоичное слово длины $p(|x|)$, выбираемое случайно по равномерному распределению, как результат работы вероятностного интерактивного алгоритма на входе вида x с собеседником G .

Последовательность стратегий $\{F_n\}$ называется неравномерно полиномиальной, если существуют последовательность схем C_{in} , где $i, n = 0, 1, 2, \dots$ такая, что размер схемы C_n ограничен полиномом от n , и функция F_n вычисляется схемой C_n . Последнее надо понимать следующим образом: Слово y нефиксированной длины подается на вход схемы в кодировке $y10^i$, где i подобрано так, что длина слова $y10^i$ равна количеству входных проводников схемы. Последовательности битовых строк закодированы словами в бинарном алфавите, например, с помощью следующего кодирования: 0 кодируется как 00, 1 кодируется как 11, а запятая как 01.

7.2 Повторение интерактивного протокола

Пусть V — интерактивный алгоритм с выходом 0,1. Пусть интерактивная стратегия P , которая общается с V , стремится максимизировать вероятность результата 1. Обозначим через $p(x)$ максимально возможную вероятности результата 1, то есть, $p(x) = \max_P \Pr[V^P(x) = 1]$.

Пусть дано произвольное натуральное число k . Рассмотрим новый интерактивный алгоритм \tilde{V} , состоящий в следующем. Будем повторять $V(x)$ последовательно k раз, используя при каждом повторении новые случайные биты. При этом стратегия P , с которой беседует \tilde{V} , не обязана действовать одинаково при каждом повторении: ее действия при очередном повторении могут зависеть от сообщений, полученных в ходе предыдущих повторений. В результате беседы у алгоритма $\tilde{V}(x)$ появится k результатов c_1, \dots, c_k . В качестве окончательного результата он выдает 1, если $c_1 = \dots = c_k = 1$, и выдает ноль иначе.

Пусть стратегия P стремится максимизировать вероятность того, что $\tilde{V}(x)$ выдаст 1. Утверждается, что при любом k оптимальная стратегия P состоит в независимом применении стратегии, оптимальной для $k = 1$, то есть,

$$\max_P \Pr[\tilde{V}^P(x) = 1] = p(x)^k.$$

Неравенство

$$\max_P \Pr[\tilde{V}^P(x) = 1] \geq p(x)^k$$

очевидно — пусть P применяет при каждом повторении оптимальную стратегию для $k = 1$. Нам нужно доказать обратное неравенство Любая стратегия P может быть разложена в композицию стратегий

$$P_1, P_2(c_1), \dots, P_k(c_1, \dots, c_{k-1}).$$

Здесь P_1 есть исходная стратегия P , оборванная после первого после первого повторения, $P_2(c_1)$ есть стратегия P , полученная из P фиксированием сообщений c_1 , полученных в ходе первого повторения, и оборванная после второго повторения, и так далее. Тогда $\Pr[\tilde{V}^P = 1]$ есть произведение вероятностей $\Pr[V^{P_1} = 1]$, $\Pr[V^{P_2(c_1)} = 1 | V^{P_1} = 1]$, \dots , $\Pr[V^{P_k(c_1, \dots, c_{k-1})} = 1 | V^{P_1} = 1, \dots, V^{P_{k-1}(c_1, \dots, c_{k-2})} = 1]$. Докажем, что каждый сомножитель здесь не превосходит p . Для этого фиксируем i . Нам надо установить, что $\Pr[V^{P_i(c_1, \dots, c_{i-1})} = 1 | V^{P_1} = 1, \dots, V^{P_{i-1}(c_1, \dots, c_{i-2})} = 1]$. Истинность условия зависит только от c_1, \dots, c_{i-1} . Поэтому достаточно доказать, что при любых фиксированных c_1, \dots, c_{i-1} Вероятность

$V^{P_i(c_1, \dots, c_{i-1})} = 1$ не превосходит p . Это следует из условия, поскольку стратегия $P_i(c_1, \dots, c_{i-1})$ не может достигнуть вероятности результата 1 большей, чем максимальная.

Если исходная стратегия P задается схемой размера s , то и все рассматриваемые стратегии $P_i(c_1, \dots, c_{i-1})$ задаются схемами не большего размера, что доказывает неравенство

$$\max_{P: \text{size}(P) \leq s} \Pr[\tilde{V}^P = 1] \leq \left(\max_{P: \text{size}(P) \leq s} \Pr[V^P = 1] \right)^k.$$

Аналогичное утверждение на самом деле верно в более общей ситуации. Будем рассматривать “игральные автоматы” следующего вида. В каждый момент функционирование автомата полностью определяется парой $\langle s, q \rangle$, где s — элемент некоторого конечного множества S , и q вершина в некотором конечном дереве Q , с корнем q_0 . Элемент s называется секретом, выбирается случайно в начале игры (по некоторому распределению), хранится в секрете (от игрока) и не меняется в ходе игры. Вершина дерева q называется позицией, известна игроку изменяется в ходе игры. Начальная позиция равна корню дерева. Ходы делаются по очереди, начиная с игрока (таким образом очередь хода в позиции определяется четностью расстояния этой позиции от корня). В свою очередь автомат выбирает один из сыновей текущей вершины дерева q по некоторому распределению, зависящему от q и секрета s . То есть, для каждой тройки s, q, q' , где q' сын вершины q неотрицательное число $p(s, q, q')$ так, что $\sum_{q'} p(s, q, q') = 1$. В свою очередь игрок выбирает также одного из сыновей текущей вершины. Игра продолжается до тех пор, пока q не станет равным некоторому листу, после чего автомат определяет, выиграл игрок или проиграл, применяя к q и секрету некоторую вероятностную функцию со значениями 0,1. То есть, для каждой пары (секрет, лист) задана вероятность $p(s, q)$ того, что игрок выиграл в соответствующей позиции.

Для каждого автомата A рассмотрим p_A — максимум по всем стратегиям игрока вероятности выиграть, пользуясь этой стратегией. Каждая пара (интерактивный алгоритм V , его вход x) очевидным образом определяет игральный автомат A такой, что $p_A = p_V(x)$. Секрет — это случайные биты автомата, а позиция состоит из слова x и последовательности сообщений, сделанных Доказывающим и Проверяющим.

Рассмотрим на автоматах операцию возведения в степень. Неформально, автомат A^k это k автоматов типа A , поставленных рядом и дей-

ствующих независимо друг от друга. Игрок может в свою очередь сделать один ход в игре с любым из k автоматов. Исходом игры является k битов. Кроме того, после окончания всех k игр, к их результатам применяется некоторая булева функция f , чтобы определить, выиграл ли игрок или проиграл. Так определенный игровой автомат будет обозначаться A_f^k .

Теперь формально. Пусть A — игральный автомат, k — произвольное натуральное число, а f — булева функция от k аргументов. Сначала построим игральный автомат A^k . Он функционирует следующим образом. Секретом является последовательность $\langle s_1, \dots, s_k \rangle$ из k секретов исходного автомата, выбираемых независимо в соответствии с распределением, входящим в определение A . Позиция в каждый момент является кортежем $\langle a_1, \dots, a_k \rangle$, где a_1, \dots, a_k — позиции исходного автомата, в которых либо во всех позициях ход игрока, либо во всех позициях, кроме одной, ход игрока. В позициях первого типа ходить должен автомат, а в позициях второго типа — игрок. Начальная позиция состоит из k начальных позиций A (и в ней очередь хода за игроком). Если в позиции $\langle a_1, \dots, a_k \rangle$ должен ходить игрок, он может выбрать любое число $i = 1, \dots, k$ и сделать разрешенный ход в игре и i -ым автоматом. То есть, новое состояние должно иметь вид $\langle a_1, \dots, a'_i, \dots, a_k \rangle$, где a'_i — сын a_i в дереве автомата A . Если в позиции $\langle a_1, \dots, a_k \rangle$ должен ходить автомат, то он делает ход, в соответствии со своей стратегией, в той из позиций a_i , где очередь хода за ним.

Листьями в новом дереве являются те кортежи, в которых все компоненты являются листьями. После того, как игра закончена, возникает k заключительных позиций и, соответственно, k результатов c_1, \dots, c_k . Это будет результатом игры автомата A^k . Результат игры автомата A_f^k получается применением к c_1, \dots, c_k функции f , то есть, игрок объявляется выигравшим, если $f(c_1, \dots, c_k) = 1$

Оказывается, что если функция f монотонна, то

$$p_{A^k} = f(p, \dots, p),$$

где $f(p_1, \dots, p_k)$ обозначает мультилинейное расширение f . Так называется функция

$$f(p_1, \dots, p_k) = \sum_{x_1, \dots, x_k \in \{0,1\}} f(x_1, \dots, x_k) \prod_i p_i^{(x_i)},$$

где $p^{(1)} = p, p^{(0)} = 1 - p$. Значение $f(p_1, \dots, p_k)$ равно вероятности того, что $f(x_1, \dots, x_k) = 1$, если каждое из x_i , независимо от других, принимает значение 1 с вероятностью p , и 0 с вероятностью $1 - p$. Нетрудно доказать, что f не убывает по каждому аргументу. Из определения непосредственно следует, что f линейна по каждой переменной.

Сначала докажем это для прозрачных автоматов. Автомат A называется прозрачным, если у него отсутствует секрет (что означает, что множество S одноэлементно). Прозрачные автоматы соответствуют играм с полной информацией, которых фиксирована вероятностная стратегия одного из игроков (самого автомата).

Ценой позиции $\langle a_1, \dots, a_k \rangle$ назовем максимум по стратегиям игрока среднего выигрыша при применении этой стратегии (среднее берется по случайным выборам, сделанным A^k). Для игр с полной информацией цену позиции можно вычислять рекурсивно: если в данной позиции очередь хода за игроком, то цена позиции есть максимум по всем возможным его ходам цены позиции, полученной в результате этого хода, а если в данной позиции очередь хода за автоматом, то цена позиции есть среднее по всем возможным его ходам цены позиции, полученной в результате этого хода.

Докажем индукцией по максимальному возможному числу оставшихся ходов, что цена позиции $c(a_1, \dots, a_k)$ равна $f(c(a_1), \dots, c(a_k))$, где $c(a)$ — цена позиции a в исходной игре (задаваемой V).

Разберем два случая:

1) Очередь хода за автоматом. Тогда цена позиции $\langle a_1, \dots, a_k \rangle$ есть среднее по всем a'_1, \dots, a'_k цены позиции $\langle a'_1, \dots, a'_k \rangle$. Здесь a'_i обозначает случайную позицию, выбранную автоматом A по распределению которое входит в его определение. По индуктивному предположению цена позиции $\langle a'_1, \dots, a'_k \rangle$ равна $f(c(a'_1), \dots, c(a'_k))$. Итак, $c(a_1, \dots, a_k)$ есть среднее значение $f(c(a'_1), \dots, c(a'_k))$. Поскольку функция f линейна по всем аргументам, среднее значение $f(c(a'_1), \dots, c(a'_k))$ совпадает со значением f на средних значениях $c(a'_1), \dots, c(a'_k)$. Поскольку среднее значение $c(a'_i)$ равно цене позиции a_i , цена позиции $\langle a_1, \dots, a_k \rangle$ равна $f(c(a_1), \dots, c(a_k))$, что и требовалось доказать.

2) Очередь хода за автоматом. Тогда цена позиции $\langle a_1, \dots, a_k \rangle$ есть максимум по всем i максимум по всем a'_i цены позиции $\langle a_1, \dots, a'_i, \dots, a_k \rangle$. По индуктивному предположению цена такой позиции есть $f(c(a_1), \dots, c(a'_i), \dots, c(a_k))$. При любом фиксированном i в силу монотонности f максимум будет достигнут при том a'_i , которое максимизирует $c(a'_i)$. То есть при любом

фиксированном i максимум будет равен $f(c(a_1), \dots, c(a_i), \dots, c(a_k))$, что не зависит от i . Так что игроку неважно, какое i выбрать, в любом случае при оптимальном выборе ходов он в среднем выиграет $f(c(a_1), \dots, c(a_k))$.

Чтобы доказать это для непрозрачных автоматов, мы переделаем любой непрозрачный автомат в эквивалентный прозрачный. Эквивалентность двух автоматов определяется так. Эквивалентные автоматы A_1, A_2 должны иметь все одинаковое, кроме множества секретов и распределения на нем, и способа выбора очередного хода автоматом (то есть вероятностных стратегий, применяемых ими). При этом для любой стратегии P игрока средний выигрыш в игре с A_1 должен быть равен среднему выигрышу в игре с A_2 .

Пусть дан непрозрачный автомат A . Новый автомат A' делает ход $q \rightarrow q'$ с вероятностью, равной условной вероятности того, что автомат A сделал бы ход $q \rightarrow q'$ при условии, что автомат A окажется в позиции q . Вероятность здесь берется по распределению на секретах и распределению, входящему в определение переходов автомата A . Аккуратно вероятность перехода хода $q \rightarrow q'$ определяется так. Зафиксируем s и определим $t(s, q)$ как вероятность того, что игрок попадет в позицию q , если будет стремиться к этому, при условии, что секрет равен s . То есть, $t(s, q)$ есть вероятность попасть в позицию q , если игрок в любой позиции, являющейся предком q выбирает ход, ведущий в q . Формально, для вершины q_{2i} четной высоты $2i$,

$$t(s, q_{2i}) = p(s, q_1, q_2) \cdot \dots \cdot p(s, q_{2i-1}, q_{2i}),$$

где q_0, q_1, \dots, q_{2i} — путь в дереве из корня в вершину q_{2i} . Для вершин q_{2i+1} нечетной высоты, $t(s, q_{2i+1}) = t(s, q_{2i})$, где q_{2i} обозначает отца q_{2i+1} .

Через $t(q)$ обозначим среднее значение вероятности попасть в вершину q , то есть $t(q) = \sum_s p(s)t(s, q)$, где $p(s)$ обозначает вероятность выбрать секрет s . Тогда вероятность перехода $q \rightarrow q'$ равна $t(q')/t(q)$. Вероятность выиграть в данном листе q определяется как условная вероятность того, что A объявил бы выигравшим игрока, если бы игра пришла в позицию q , то есть,

$$\sum_s p(s)t(s, q)p(s, q)/t(q),$$

где $p(s, q)$ обозначает вероятность того, что A объявил игрока выигравшим в позиции q . Вероятности переходов в новом автомате подобраны

таким образом, что для него вероятность попасть в позицию q , если игрок стремится к этому, равна $t(q)$.

Пусть зафиксирована некоторая стратегия игрока. Иными словами в дереве позиций для всех вершин четной высоты выбран только один сын, а все остальные сыновья вместе с их потомками уничтожены. Тогда вероятность выигрыша в исходной игре определяется как сумма по всем s и всем листьям q урезанного дерева величины $p(s)t(s, q)p(s, q)$. А в новой игре — как сумма по всем листьям q урезанного дерева величины

$$t(q) \frac{\sum_s p(s)t(s, q)p(s, q)}{t(q)}.$$

Если произвести сокращение, то станет ясно, что эти вероятности равны.

Вспомним, что мы собирались доказать равенство

$$p_{A_f^k} = f(p_A, \dots, p_A),$$

для непрозрачных автоматов A . Пусть \tilde{A} обозначает эквивалентный A прозрачный автомат. Нетрудно проверить, что автоматы \tilde{A}^k и \tilde{A}^k совпадают (имеют одинаковые вероятности всех переходов). Поэтому автомат \tilde{A}^k эквивалентен автомату A^k . А следовательно

$$p_{A_f^k} = p_{\tilde{A}_f^k} = f(p_{\tilde{A}}, \dots, p_{\tilde{A}}) = f(p_A, \dots, p_A).$$

Заметим, что если исходный игровой автомат задавался полиномиальным вероятностным алгоритмом V , то новый уже не будет, поскольку для него вероятности переходов $q \rightarrow q'$ уже не вычислимы за полиномиальное время. Однако он задается алгоритмом, работающим на полиномиальной зоне, поскольку вероятность каждого перехода $q \rightarrow q'$ можно найти перебором, и то же самое относится к вероятности выигрыша в данной позиции. Поскольку цену позиции в игре с полной информацией можно находить рекурсивно на полиномиальной зоне, это доказывает включение IP в PSPACE.

Автомат \tilde{V} имеет k каналов общения с игроком и для каждого канала применяет стратегию V . Для разных каналов стратегия V применяется независимо. Можно себе представлять \tilde{V} как вероятностную стратегию, играющую параллельно и независимо друг от друга k партий в игру, соответствующую V . В каждый момент состояние автомата является кортежем из k состояний исходного автомата V и еще одним битом, определяющим очередь хода. Начальная позиция — это кортеж, состоящий из

k начальных состояний V и очередь хода принадлежит игроку. В свою очередь стратегия \tilde{V} в каждой из партий, в которой ее ход, делает выборы, предписываемые стратегией V и сообщает игроку соответствующую информацию. Для разных партий случайные эти случайные величины независимы. То есть ходы стратегии \tilde{V} в разных партиях не зависят друг от друга.

Игроку в его очередь, разрешается сделать по одному ходу в любых партиях, в которых его ход, но не менее чем в одной (иначе игра может никогда не закончиться).

Если Доказывающий обязан сделать ход во всех позициях, то такая игра называется параллельным повторением V . Если сначала нужно сыграть первую игру, затем вторую и так далее, то такая игра называется последовательным повторением V .

7.3 Интерактивные протоколы привязки

Интерактивный протокол привязки к биту состоит из трех алгоритмов: вероятностных полиномиальных интерактивных алгоритмов S, R и детерминированного полиномиального алгоритма T . Алгоритм S получает на вход параметр безопасности n и один бит σ , а алгоритм R получает на вход только n . После этого эти алгоритмы общаются друг с другом, то есть, по очереди посылают друг другу сообщения. Это общение называется стадией привязки или стадией закрытия сундучка. Через $c(S_\sigma, R)$ мы будем обозначать последовательность всех переданных сообщений между S и R , если S получил на вход σ . Последовательность $c(S_\sigma, R)$ является случайной величиной, причем зависящей от параметра безопасности (который мы для краткости опускаем в обозначениях). Она выполняет роль привязки в неинтерактивных протоколах. Кроме того, алгоритм S выдает на выход строку $k(S_\sigma, R)$, называемую ключом, которая может зависеть от сообщений, полученных от R (поэтому мы включаем R в обозначения). Алгоритм T получает на вход n , строку c и строку k и выдает на выход один бит или сообщение о том, что один из двух игроков блокировал протокол. Будем эти сообщения обозначать \perp_S, \perp_R (\perp_S означает, что протокол блокирован посылающим, а \perp_R — получателем).

Требования к этим алгоритмам такие.

Первое требование заключается в следующем. Для любого бита σ с

вероятностью, приблизительно равной 1, выполнено

$$T(c(S_\sigma, R), k(S_\sigma, R)) = \sigma. \quad (2)$$

Более того, это требование должно быть выполнено, даже если, получатель жульничает. Уточним, что это означает.

Пусть получатель на стадии закрытия сундучка выполняет не алгоритм R , а какой-то другой алгоритм R^* . При этом он хочет добиться того, чтобы при его открытии появился противоположный запечатанному бит. Требуется, чтобы это было невозможно, если R^* вычисляется схемами полиномиального от n размера. Итак, уточненное требование выглядит так.

(1) Для любой неравномерно полиномиально последовательности стратегий $\{R_n^*\}$ с вероятностью, приблизительно равной 1 (при $n \rightarrow \infty$), для любого бита σ выполнено

$$T(c(S_\sigma, R^*), k(S_\sigma, R^*)) \in \{\sigma, \perp_R\}.$$

(для краткости мы опускаем параметр безопасности n в обозначениях).

(2) Представим, что отправитель хочет обмануть получателя: на стадии закрытия сундучка он выполняет не алгоритм S , а неравномерно полиномиальную стратегию S^* , которая после окончания беседы выдает две строки, которые мы будем обозначать через $k_0 = k_0(S^*, T)$ и $k_1 = k_1(S^*, T)$. При этом он хочет добиться, чтобы для появившегося диалога $c(S^*, T)$ было одновременно выполнено $T_n(c, k_0) = 0$ и $T_n(c, k_1) = 1$. Требуется, чтобы это было невозможно, если стратегия S^* вычисляется схемой полиномиального (от n) размера: для любой неравномерно полиномиальной последовательности стратегий $\{S_n^*\}$ (выдающей после стадии привязки два ключа) для всех n вероятность события

$$T(c(S^*, R), k_0(S^*, R)) = 0 \quad \text{и} \quad T(c(S^*, R), k_1(S^*, R)) = 1$$

пренебрежима мала. Кроме того, требуется, чтобы никакая неравномерно полиномиальная стратегия не могла заставить судью признать блокирующим честного получателя: для неравномерно полиномиальной последовательности стратегий $\{S_n^*\}$ (выдающей после стадии привязки один ключ) для всех n вероятность события

$$T(c(S^*, R), k(S^*, R)) = \perp_R$$

пренебрежимо мала. Вместе с условием (1) это гарантирует, что для всех σ с приблизительно единичной вероятностью выполнено равенство (2).

Иногда вместо условия (2) требуется более сильное условие:

(2') Для любой (не обязательно неравномерно полиномиальной) последовательности стратегий $\{S_n^*\}$ для всех n вероятность того, что существуют k_0, k_1 , для которых

$$T(c(S^*, R), k_0) = 0 \quad \text{и} \quad T(c(S^*, R), k_1) = 1$$

пренебрежимо мала, а также вероятность события

$$\exists k T(c(S^*, R), k) = \perp_R$$

пренебрежимо мала.

В последнем требовании мы заботимся об интересах отправителя. Допустим, жульничает получатель, и вместо алгоритма R использует неравномерно полиномиальную последовательность стратегий R_n^* . Требуется, чтобы возникающая привязка не несла никакой информации о сообщении.

(3) Для любой неравномерно полиномиальной последовательности стратегий $\{R_n^*\}$ случайные величины $c(S_0, R^*)$ и $c(S_1, R^*)$ вычислительно неотличимы.

Если выполнены все три условия, то тройка алгоритмов R, S, T называется интерактивным протоколом привязки к биту.

Задача 33. Докажите, что условие (2') можно переформулировать так: существует функция $b(c)$ со значениями 0,1 такая, что для любой последовательности стратегий $\{S_n^*\}$ с приблизительно единичной вероятностью для всех k выполнено $T(c(S^*, R), k) \in \{b(c(S^*, R)), \perp_S\}$. Неформально, $b(c)$ есть тот бит, который закрыт в сундучке. Любой ключ k либо открывает этот бит, либо не подходит к замку.

Теорема 27. Если существует генератор ПСЧ, то существует интерактивный протокол привязки к биту. Для этого протокола второе условие выполнено в усиленном варианте, то есть, выполнено (2').

Доказательство. Пусть G генератор ПСЧ $\{0, 1\}^n \rightarrow \{0, 1\}^{3n}$. Алгоритм R посылает случайную строку r длины $3n$, после чего останавливается. Алгоритм S ждет от R строку r длины $3n$. Если присланная строка имеет другую длину, то он заканчивает диалог (посылая пустое слово).

Иначе, он выбирает случайную строчку s длины n и посылает $G(s)$, если $\sigma = 0$, и посылает $G(s) \oplus r$ иначе (то есть, он посылает $G(s) \oplus \sigma \cdot r$). Ключ, выдаваемый алгоритмом S есть σs .

Входом алгоритма T , кроме n , является пара строк r, x и строка k . Если все строки имеют правильную длину, причем $x = G(s) \oplus \sigma \cdot r$, где σ первый бит строки k , а s — остальные ее биты, то алгоритм T выдает σ . Иначе, алгоритм T разбирается, кто заблокировал протокол: если длина r не равна $3n$, то заблокировал получатель. Если длина r равна $3n$, но длина k не равна $n+1$ или $x \neq G(s) \oplus \sigma \cdot r$, то заблокировал отправитель.

Ясно, что условие (1) и вторая часть условия (2') выполнены. Проверим первую часть условия (2'). Пусть дана последовательность стратегий S_n^* . Алгоритм T получает на вход ключ k и диалог, который состоит из строк $r, S^*(r)$. Допустим, алгоритм T , получив на вход ключи k_0, k_1 и диалог $r, S^*(r)$, выдает соответственно 0,1. Ясно, что тогда $k_0 = 0s_0$ и $k_1 = 1s_1$ для некоторых строк s_0, s_1 длины n . Поэтому достаточно доказать, что вероятность события

$$(\exists s_0, s_1) \quad S^*(r) = G(s_0), \quad S^*(r) = G(s_1) \oplus r$$

пренебрежимо мала. Вероятность этого события не больше вероятности того, что найдутся s_0, s_1 такие, что $G(s_0) = G(s_1) \oplus r$. При фиксированных s_0, s_1 вероятность этого события равна 2^{-3n} . Суммируя по всем s_0, s_1 , получаем не больше 2^{-n} , что стремится к нулю быстрее любого обратного полинома от n .

Проверим условие (3). Пусть дана последовательность схем R_n^* полиномиального от n размера, которые выдают на выход одну строку $3n$, не имея входа. Другими словами R_n^* просто задает последовательность строк r_n^* . При тех n , для которых длина r_n^* не равна $3n$, диалог имеет вид $\langle r_n^*, \text{пустое слово} \rangle$, и не зависит от σ . При остальных n нам надо доказать вычислительную неотличимость случайных величин $\langle r_n^*, G(s) \rangle$ и $\langle r_n^*, G(s) \oplus r_n^* \rangle$. Они вычислительно неотличимы друг от друга, поскольку обе вычислительно неотличимы от случайной величины $\langle r_n^*, \gamma_{3n} \rangle$, где γ_{3n} равномерно распределенная среди слов длины $3n$ случайная величина. \square

Задача 34. Докажите, что без ограничения общности можно считать, что интерактивный протокол привязки к биту в качестве ключа всегда выдает свои случайные биты и σ .

Аналогичным образом можно определить понятие протокола привязки к одному из чисел $\{1, \dots, m\}$ где m — константа или растет с ростом параметра безопасности с экспоненциальной скоростью: $m = 2^{\text{poly}(n)}$. А именно, как и раньше протоколом привязки называется тройка алгоритмов $\langle R, S, T \rangle$, для которых выполнены условия

(1) Для любой неравномерно полиномиально последовательности стратегий $\{R_n^*\}$ с вероятностью, приблизительно равной 1 (при $n \rightarrow \infty$), для любого $i \in \{1, \dots, m\}$ выполнено

$$T(c(S_i, R^*), k(S_\sigma, R^*)) \in \{\sigma, \perp_R\}.$$

(для краткости мы опускаем параметр безопасности n в обозначениях).

(2) Для любой неравномерно полиномиальной последовательности стратегий $\{S_n^*\}$ (выдающих в конце стадии привязки два ключа) для всех n вероятность события “ $T(c(S^*, R), k_0(S^*, R))$ и $T(c(S^*, R), k_1(S^*, R))$ различные элементы множества $\{1, \dots, m\}$ ” пренебрежима мала. Для любой неравномерно полиномиальной последовательности стратегий $\{S_n^*\}$ (выдающих в конце стадии привязки один ключ) для всех n вероятность события

$$T(c(S^*, R), k(S^*, R)) = \perp_R$$

пренебрежима мала.

(2') Для любой последовательности стратегий $\{S_n^*\}$ для всех n вероятность того, что существуют k_0, k_1 , для которых $T(c(S^*, R), k_0)$ и $T(c(S^*, R), k_1)$ различные элементы множества $\{1, \dots, m\}$, пренебрежима мала, а также вероятность события

$$\exists k T(c(S^*, R), k) = \perp_R$$

пренебрежима мала.

(3) Для любой неравномерно полиномиальной последовательности стратегий $\{R_n^*\}$ случайные величины $c(S_i, R^*)$ для $i = 1, \dots, m$ вычислительно неотличимы.

Задача 35. Докажите, что условие (2') можно переформулировать так: существует функция $K(c)$ со значениями в $\{1, \dots, m\}$ такая, что для любой последовательности стратегий $\{S_n^*\}$ с приблизительно единичной вероятностью для всех k выполнено $T(c(S^*, R), k) \in \{K(c(S^*, R)), \perp_S\}$.

Протокол привязки к элементам множества $\{1, \dots, m\}$ можно построить из любого протокола (S, R, T) привязки к одному биту. Сделаем это

для случая $m = 2^{p(n)}$, где p — некоторый полином (для случая постоянного m конструкция аналогична). Числа из $\{1, \dots, m\}$ отождествим с битовыми строками длины $p(n)$. Посылающий и получающий привязываются по очереди к каждому из битов этой строки. Другими словами, новая стратегия \tilde{S} есть $p(n)$ -кратное последовательное повторение стратегии S , новая стратегия \tilde{R} есть $p(n)$ -кратное последовательное повторение стратегии R , а новый алгоритм раскрытия сундучка есть $p(n)$ -кратное применение старого алгоритма T . Если хотя бы в одном из применений T выдал \perp_S или \perp_R , то новый алгоритм выдает, соответственно, \perp_S или \perp_R (если в одном из применений было выдано \perp_S , а в другом \perp_R , то неважно, что выдать). Иначе надо выдать полученную битовую строку.

Истинность условий (1), (2), (3) (а также свойства (2')), если исходный протокол удовлетворял условию (2')) следуют из свойств последовательного повторения интерактивного протокола.

8 Протоколы бросания монетки по телефону

Протоколом бросания монетки по телефону называется тройка алгоритмов A, B, J . Каждый из этих алгоритмов получает на вход натуральное число n в унарной записи и останавливается, проработав полиномиальное от n число шагов. Кроме того, алгоритм J получает на вход еще и двоичную строку c . Алгоритмы A, B являются вероятностными интерактивными алгоритмами без выхода, а алгоритм J детерминированным неинтерактивным алгоритмом, выдающим на выход A или B .

Эти три алгоритма следующим образом используются для игры в орлянку по телефону. Чтобы подбросить монетку, игроки, Алиса и Боб, выполняют алгоритмы A, B соответственно. Эти алгоритмы общаются по телефону (без ограничения общности можно считать, что первое сообщение посылается алгоритмом A). Запись их общения $c(A, B)$ дается на вход алгоритму J и он определяет, кто выиграл: если $J(c(A, B)) = A$, то выиграла Алиса, а если $J(c(A, B)) = B$, то выиграл Боб.

Требуется, чтобы даже если Боб жульничает, и вместо алгоритма B запускает некоторую неравномерно полиномиальную стратегию B_n^* , то вероятность того, что Алиса проиграет превосходить $1/2$ (вероятность проиграть при обычной игре в орлянку) не более, чем на пренебрежимо малую величину (точнее, на величину, стремящуюся к нулю быстрее любого обратного полинома). Аналогичное требование должно

быть выполнено, и для вероятности проигрыша Боба, если жульничает Алиса.

Формально, должны быть выполнены следующие два требования.

(1) Для любой неравномерно полиномиальной стратегии B_n^* вероятность события $J(c(A, B^*)) = B$ не превосходит $1/2 + \alpha_n$, где α_n пренебрежимо мало (стремится с ростом n к нулю быстрее любого обратного полинома).

(2) Для любой неравномерно полиномиальной стратегии A_n^* вероятность события $J(c(A^*, B)) = A$ не превосходит $1/2 + \beta_n$, где β_n пренебрежимо мало.

Замечание 1. Важно, что алгоритм J детерминирован. Иначе, задача была бы тривиальной, поскольку алгоритм J сам бы мог подбросить монетку и объявить победителя, исходя из результата бросания. Кроме того, пользы от вероятностного алгоритма никакой нет, поскольку игроки не могут запустить его из-за отсутствия общего источника случайности.

Замечание 2. Алгоритм J задает некоторую игру G_n (одну для каждого значения параметра безопасности) с конечным числом ходов. А именно, количество ходов в этой игре и длина каждого хода ограничено полиномом $p(n)$, ограничивающим время работ алгоритмов A и B . Игроки ходят по очереди, начиная с Алисы. Последовательность сделанных ходов дается на вход алгоритму J , который и определяет, кто выиграл. По теореме Кенига, в любой игре с конечным числом ходов один из игроков имеет в этой игре выигрышную стратегию. Если для бесконечно многих n этим игроком является Алиса, то условие (2) не выполнено для выигрышной стратегии Алисы (для тех n , для которых такая существует): вероятность ее выигрыша равна 1 для этих n . Аналогично, если для бесконечно многих n этим игроком является Боб, то условие (1) не выполнено для некоторой стратегии Боба. Поэтому в обоих условиях (1) и (2) важно, что мы ограничиваем вычислительные возможности жульничающего игрока.

Теорема 28. *Если существует интерактивный протокол привязки к биту, то существует и протокол подбрасывания монетки по телефону.*

Доказательство. Пусть нам дан некоторый протокол (S, R, T) привязки к биту.

Алгоритм A (выполняемый Алисой) действует так: подбрасываем монетку один раз и получаем случайный бит, который будем обозначать

через σ . Запускаем алгоритм $S(\sigma)$, который будет общаться с Бобом. Когда общение закончится и алгоритм $S(\sigma)$ выдаст некоторый ключ k , мы останавливаемся и ждем, пока нам не пришлют следующее сообщение. После этого посылаем ключ k .

Алгоритм B (выполняемый Бобом) действует так: запускаем алгоритм R , который будет общаться с Алисой. Когда общение закончится, подбрасываем монетку и посылаем результат бросания τ Алисе.

Алгоритм J действует так. Пусть c обозначает протокол общения на первой стадии. (Чтобы можно было отделить в протоколе, где заканчиваются сообщения первой стадии, алгоритмы S и R надо модифицировать так, чтобы они приписывали ко всем сообщениям 0 слева. На второй стадии все сообщения будут начинаться на 1.) Запускаем алгоритм T на паре c, k , где k последнее сообщение Алисы. Объявляем выигравшим Боба, если T выдает, сообщение, что протокол привязки заблокирован Алисой, или на второй стадии Боб послал бит, совпадающий с битом, выданным T .

Проверим, что этот протокол соблюдает интересы обоих игроков, то есть, условия (1) и (2).

Интересы Алисы. Пусть Алиса действует честно, а Боб руководствуется некоторой неравномерно полиномиальной стратегией B_n^* . Это означает, что на первой стадии он выполняет некоторую неравномерно полиномиальную стратегию R_n^* , а на второй стадии применяет к протоколу беседы c на первой стадии некоторую схему τ_n^* полиномиального от n размера. Оценим сверху вероятность того, что алгоритм J объявит проигравшей Алису. Такое может произойти в двух случаях: алгоритм T объявит Алису заблокировавшей протокол и алгоритм T выдаст бит, равный $\tau^*(c(S(\sigma), R^*))$. Поскольку Алиса не жульничает, в силу первого требования к протоколу привязки вероятность первого события пренебрежимо мала. В силу того же требования вероятность второго события приблизительно равна вероятности того, что $\tau^*(c(S(\sigma), R^*))$ равно σ . Так как Алиса выбирает свой бит σ случайно, вероятность этого события равна

$$\frac{\Pr[\tau^*(c(S(0), R^*)) = 0] + \Pr[\tau^*(c(S(1), R^*)) = 1]}{2}.$$

В силу второго требования к протоколу привязки случайные величины $c(S(0), R^*)$ и $c(S(1), R^*)$ вычислительно неотличимы. В частности, они неотличимы и схемой τ^* . Поэтому, если во втором слагаемом заменить вторую случайную величину на первую, вероятность изменится на пре-

небрежимо малую величину. После такой замены в числителе дроби

$$\frac{\Pr[\tau^*(c(S(0), R^*)) = 0] + \Pr[\tau^*(c(S(0), R^*)) = 1]}{2}$$

мы получим сумму вероятностей взаимно исключающих событий, поэтому числитель не превосходит 1, а значит вся дробь не превосходит $1/2$.

Интересы Боба. Пусть Боб действует честно, а Алиса руководствуется некоторой неравномерно полиномиальной стратегией A_n^* . Это означает, что на первой стадии она выполняет некоторую неравномерно полиномиальную стратегию S_n^* , а на второй стадии применяет к протоколу беседы c , и биту τ , посланному Бобом, некоторую схему k_n^* полиномиального от n размера. Оценим сверху вероятность того, что алгоритм J объявит проигравшей Боба. Такое может произойти в двух случаях: алгоритм T объявит Боба блокировавшим протокол и алгоритм T выдаст бит, отличный от τ , то есть $\bar{\tau}$.

Поскольку Боб действует честно, по третьему требованию вероятностью первого события можно пренебречь. Поскольку свой бит τ Боб выбирает случайно, вероятность второго события равна

$$\frac{\Pr[T(c(S^*, R), k^*(c(S^*, R), 0)) = 1] + \Pr[T(c(S^*, R), k^*(c(S^*, R), 1)) = 0]}{2}.$$

Оценим сверху вероятность этого события, используя оценку

$$\Pr[U] + \Pr[V] = \Pr[U \cup V] + \Pr[U \cap V] \leq 1 + \Pr[U \cap V],$$

верную для любых двух событий U, V . В числителе нашей дроби суммируются вероятности двух событий U, V . Вероятность их пересечения пренебрежимо мала, поскольку третье требование к протоколу привязки гарантирует, что вероятность существования ключей k_0, k_1 , для которых

$$T(c(S^*, R), k_0) = 0, \quad T(c(S^*, R), k_1) = 1,$$

пренебрежимо мала. Поэтому дробь превосходит $1/2$ на пренебрежимо малую величину. \square

Задача 36. Пусть в качестве протокола привязки в этой схеме использован неинтерактивный протокол, основанный на однозначной односторонней функции. У кого из двух игроков есть выигрышная стратегия в полученной игре (без ограничения на равномерную полиномиальность)?

Задача 37. Пусть в качестве протокола привязки в этой схеме использован интерактивный протокол из предыдущего раздела, основанный на существовании генератора ПСЧ. У кого из двух игроков есть выигрышная стратегия в полученной игре (без ограничения на равномерную полиномиальность)?

9 Интерактивные доказательства с нулевым раскрытием

9.1 Интерактивные доказательства

Скажем, что множество слов L принадлежит классу IP , если существует полиномиальный вероятностный интерактивный алгоритм V с таким двумя свойствами.

- (1) Для всех $x \in L$ существует стратегия P такая, что $\Pr[V^P(x) = 1] \approx 1$.
- (2) Для всех $x \notin L$ для любой стратегии P выполнено $\Pr[V^P(x) = 1] \approx 0$.

Вероятность в обоих случаях берется по случайным битам алгоритма V , так что стоило бы писать $V^P(x, r)$, а не $V^P(x)$. Приблизительное равенство в обоих требованиях понимается при стремлении $n = |x|$ к бесконечности. Обычно алгоритм V называется Проверяющим, а стратегия P — Доказывающим.

Теорема 29. Язык GI , состоящий из множества всех пар $\langle G_0, G_1 \rangle$ неизоморфных графов, принадлежит IP .

Мы считаем, что множества вершин графов G_0, G_1 совпадают и равны множеству чисел $\{1, 2, \dots, n\}$ для некоторого n . Оба графа задаются матрицей смежности, которая, в свою очередь, задается $n(n-1)/2$ битами.

Доказательство. Стратегия Проверяющего заключается в следующем. Выбираем случайно один бит α (оба значения 0,1 с вероятностями $1/2$). Выбираем случайную перестановку π множества вершин $\{1, 2, \dots, n\}$ (все перестановки равновероятны). Переставляем в графе G_α вершины с помощью перестановки π и посылаем полученный граф πG_α (точнее, его матрицу смежности). Если в ответ Доказывающий прислал что-то, отличное от бита α , то заканчиваем общение досрочно. Иначе по повторяем

то же самое, используя новый бит α и новую перестановку π . И так далее, n раз. Если не произошло досрочной остановки, то выдаем 0, а иначе 1.

Ясно, что если графы G_0, G_1 не изоморфны, то у Доказывающего имеется стратегия P такая, что с вероятностью 1 выполнено $V^P(G_0, G_1) = 1$. Пусть графы неизоморфны. Докажем, что вероятность того, что в первом раунде Доказывающий прислал α , не больше $1/2$. Действительно, поскольку мы выбираем оба бита 0, 1 равновероятно, вероятность этого события равна

$$\frac{\Pr[P(\pi G_0) = 0] + \Pr[P(\pi G_1) = 1]}{2}.$$

Поскольку графы G_0, G_1 изоморфны, случайные величины $\pi G_0, \pi G_1$ имеют одинаковое распределение, поэтому в первом слагаемом в этой сумме πG_0 можно заменить на πG_1 (вероятность от этого не изменится). В результате мы получим полусумму вероятностей непересекающихся событий, которая не может превосходить $1/2$.

Рассуждая таким образом, легко доказать по индукции, что после выполнения k раундов вероятность того, что мы не остановимся досрочно не больше 2^{-k} . Чтобы сделать индуктивный переход, достаточно установить, что при любых значениях $\alpha_1, \dots, \alpha_k, \pi_1, \dots, \pi_k$ вероятность того, что в $k + 1$ раунде Доказывающий правильно найдет α_{k+1} при условии, что в предыдущих раундах Проверяющий выбрал $\alpha_1, \dots, \alpha_k, \pi_1, \dots, \pi_k$, не превосходит $1/2$. Это доказывается так. Ход, делаемый Доказывающим в раунде $k + 1$ зависит от наших ходов в предыдущих k раундах и от хода πG_α сделанного нами в раунде $k + 1$. Поскольку мы предполагаем $\alpha_1, \dots, \alpha_k, \pi_1, \dots, \pi_k$, фиксированными, ход Доказывающего зависит только от $\pi(G_\alpha)$. Поэтому годится то же рассуждение, что и для первого раунда. \square

9.2 Стратегии с нулевым разглашением

Пусть $F_{x,n}$ полиномиально вычислимая вероятностная стратегия с параметрами $x, 1^n$ (где n — параметр безопасности). Пусть $f_n(x)$ некоторая полиномиально вычислимая функция от x и 1^n .⁷ В дальнейшем мы будем везде, где только возможно, опускать параметр безопасности

⁷Можно и рассматривать не полиномиально вычислимые стратегии и функции. Для дальнейшего важно лишь, что длина выхода F, f ограничена полиномом от длины входа.

в обозначениях. Мы говорим, что стратегия F_x не разглашает ничего, кроме, возможно, $f(x)$ (то есть, разглашает только $f(x)$), если существует полиномиальный вероятностный алгоритм S с оракулом, для которого выполнено следующее. Для любой последовательности слов x_n полиномиальной от n длины и любой неравномерно полиномиальной последовательности стратегий $\{G_n\}$, алгоритм S , получив на вход $1^n, f(x)$, и используя в качестве оракула G_n , генерирует некоторую случайную величину $S^G(f(x_n))$, вычислительно неотличимую от случайной величины $c(F_{x_n}, G_n)$ (напомним, что $c(F_x, G)$ обозначает запись беседы между стратегиями F_x и G). В частности, случайные величины $c(F(x), G)$ при разных x с одинаковым значением $f(x)$ вычислительно неотличимы друг от друга. Алгоритм S будем называть симулятором. В этом определении важно, что симулятору разрешается не просто общаться со стратегией G , а можно запрашивать значение G на произвольных словах. Будем говорить, что F_x имеет нулевое разглашение, если она разглашает только $f(x) \equiv (\text{пустое слово})$.

В определении неразглашения стратегия G предполагается детерминированной. Для вероятностных стратегий G определение такое. Стратегию, полученную из G фиксацией случайных битов r , будем обозначать через $G(r)$. Стратегия F_x не разглашает ничего, кроме, возможно, $f(x)$ (то есть, разглашает только $f(x)$), если существует полиномиальный вероятностный алгоритм S с оракулом G (оракул по данным r, y сообщает значение $G(r, y)$), для которого выполнено следующее. Для любой последовательности слов x_n полиномиальной от n длины и любой неравномерно полиномиальной последовательности вероятностных стратегий $\{G_n\}$, алгоритм S , получив на вход $1^n, f(x)$, и используя в качестве оракула G_n , генерирует некоторую случайную величину $S^G(f(x_n))$, вычислительно неотличимую от случайной величины $\langle r, c(F_x, G(r)) \rangle$.

Полученное определение эквивалентно исходному. Пусть имеется неравномерно полиномиальная последовательность вероятностных стратегий $\{G_n\}$. По условию количество случайных битов G_n есть полином от n , и слово, выдаваемое стратегией $G(r)$ на последовательности сообщений y , можно вычислить по r, y схемой полиномиального от n размера со входами r, y . Чтобы породить случайную, вычислительно неотличимую от случайной величины $\langle r, c(F_x, G(r)) \rangle$, мы выберем r случайным образом, запустим алгоритм S с оракулом $G(r)$ и затем выдадим на выход результат его работы, спаренный с r .

Определение неразглашения монотонно в следующем смысле: пусть

$f(x), g(x)$ произвольные функции (зависящие еще от параметра безопасности). Пусть стратегия F_x разглашает только $f(x)$. Тогда она также не разглашает ничего, кроме $\langle f(x), g(x) \rangle$, что очевидно.

Еще одно свойство состоит в следующем. Допустим, что некоторая информация $g(x)$ о секретном x общеизвестна (например, общеизвестным может быть само $f(x)$). В этом случае естественно хотеть, чтобы из записи беседы противник не мог извлечь более информации, чем имеется в $f(x)$, даже используя $g(x)$. При этом следует предполагать, что противник беседуя с F_x использует некоторую неравномерно полиномиальную стратегию $G_{g(x)}$, которой известно $g(x)$. А значит, он получает от F_x информацию $u = c(F_x, G_{g(x)})$. Скажем, что F_x разглашает только $f(x)$ при общеизвестном $g(x)$, если найдется полиномиальный вероятностный алгоритм S такой, что для любой последовательности слов x_n полиномиальной от n длины и любой неравномерно полиномиальной стратегии $G_{n,g(x_n)}$, случайные величины $\langle \tilde{u}_n, g(x_n) \rangle$ и $\langle u_n, g(x_n) \rangle$ вычислительно неотличимы, где $u_n = c(F_{x_n}, G_{g(x_n)})$ и $\tilde{u}_n = S^{G_{g(x_n)}}(f(x_n), g(x_n))$. Заметим, что в этом определении $G_{n,g(x_n)}$ можно заменить на G_n , поскольку x_n уже зафиксировано. По этой же причине неотличимость величин $\langle \tilde{u}_n, g(x_n) \rangle$ и $\langle u_n, g(x_n) \rangle$ равносильна неотличимости величин \tilde{u}_n и u_n . Итак, мы приходим к следующему определению.

Определение. Стратегия F_x разглашает только $f(x)$ при общеизвестном $g(x)$, если найдется полиномиальный вероятностный алгоритм S такой, что для любой последовательности слов x_n полиномиальной от n длины и любой неравномерно полиномиальной стратегии G_n , случайные величины $c(F_{x_n}, G_n)$ и $S^{G_n}(f(x_n), g(x_n))$ вычислительно неотличимы.

Нетрудно однако понять, что это определение эквивалентно неразглашению в отсутствие общеизвестной информации.

Лемма 30. Если F_x разглашает только $f(x)$, то для любой полиномиально вычислимой функции $g_n(x)$ стратегия F_x разглашает только $f(x)$ при общеизвестном $g(x)$.

Доказательство. Нам известно, что для стратегии F_x имеется симулятор S . Изготовим из него новый симулятор $\tilde{S}^G(f(x), g(x)) = S^{G_{g(x)}}(f(x))$. Если для какой-нибудь последовательности слов x_n полиномиальной длины, случайные величины $S^{G_{g(x_n)}}(f(x_n))$ и $c(F_{x_n}, G_{g(x_n)})$, то означает, что симулятор S ошибается на неравномерно полиномиальной стратегии $G_{g(x_n)}$. \square

Мы уже встречались с понятием неразглашения в частных случаях. Пусть (K, E, D) — произвольная схема шифрования с закрытым ключом. Рассмотрим следующую стратегию $F_{n,x}$: она выбирает случайно ключ e (используя алгоритм K) и сообщает шифрограмму $E_n(e, x)$, после чего останавливает общение. Одним из требований на схему шифрования было такое: для любой последовательности слов x_n полиномиальной длины случайная величина $E(e, x_n)$ вычислительно неотличима от случайной величины $E(e, y_n)$, где y_n — слово из одних нулей той же длины, что и x_n . В новой терминологии это означает, что стратегия $F_{n,x}$ разглашает только $|x|$.

Другой пример: в определении протокола привязки к биту требуется, чтобы вероятностная стратегия F_σ (где $\sigma = 0, 1$), которая сообщает привязку $c(\sigma, r)$ (где r выбирается случайно) и затем останавливается, имела нулевое разглашение.

Установим одно важное свойство понятия неразглашения. Для этого определим операцию последовательного выполнения (композиции) стратегий. Пусть F вероятностная стратегия. Выполним сначала стратегию F , а затем, забыв все сделанные нами и собеседником сообщения, выполним ее еще раз. Иными словами, результат применения новой стратегии к последовательности ходов c_1, c_2, \dots, c_{2k} определяется так: если не существует такого i , что $F(c_1, c_2, \dots, c_{2i})$ есть пустое слово (то есть, стратегия F не остановила диалог), то выдаем $F(c_1, c_2, \dots, c_{2k})$. Иначе выдаем $F(c_{2i+1}, \dots, c_{2k})$, где i наименьшее такое, что $F(c_1, c_2, \dots, c_{2i})$ — пустое слово. Полученную стратегию обозначим через F^2 .

Пусть F_x — вероятностная стратегия с параметром x . Рассмотрим стратегию с параметром x , заключающуюся в двукратном последовательном выполнении стратегии F_x . Полученную стратегию обозначим через F_x^2 .

Лемма 31. *Если стратегия F_x разглашает только $f(x)$, то стратегия F_x^2 разглашает только $f(x)$.*

Доказательство. Зафиксируем симулятор S для стратегии F_x и построим симулятор \tilde{S} для стратегии F_x^2 . Новый симулятор должен, получив на вход $y = f(x)$ и имея в качестве оракула некоторую неравномерно полиномиальную стратегию G , сгенерировать случайную величину, вычислительно неотличимую от $c(F_x^2, G)$ (мы будем опускать в дальнейшем индекс n).

После того, как стратегия F_x останавливается, в диалоге между F_x^2 и G сделаны какие-то сообщения c_1, \dots, c_{2k} (последнее сообщение c_{2k+1} сделано стратегией F_x и является пустым словом). Мы можем считать, что со второй копией F_x беседует стратегия $G_{c_1, \dots, c_{2k}}$, определяемая равенством

$$G_{c_1, \dots, c_{2k}}(c_{2k+1}, c_{2k+2}, \dots) = G(c_1, c_2, \dots, c_{2k}, c_{2k+1}, c_{2k+2}, \dots).$$

Поэтому диалог стратегий F_x^2 и G равен $\langle u, v(u) \rangle$, где u — диалог стратегий F_x и G , а $v(u)$ — диалог стратегий F_x и G_u . Чтобы сгенерировать эту случайную величину, запускаем симулятор S на входе y с оракулом G . Он выдаст некоторую последовательность сообщений $\tilde{u} \sim u$. Запустим опять симулятор S с оракулом $G_{v(\tilde{u})}$ на входе y , используя новые случайные биты. Он нам выдаст последовательность $\tilde{v}(\tilde{u}) \sim c(F_x, G_{v(\tilde{u})})$.

Нам надо доказать вычислительную неотличимость случайных величин

$$\xi_0 = \langle u, v(u) \rangle.$$

и

$$\xi_2 = \langle \tilde{u}, \tilde{v}(\tilde{u}) \rangle.$$

Достаточно доказать, что они обе неотличимы от случайной величины

$$\xi_1 = \langle u, \tilde{v}(u) \rangle.$$

Неотличимость ξ_0 и ξ_1 доказывается так. По условию при любом фиксированном значении u случайные величины $v(u)$ и $\tilde{v}(u)$ вычислительно неотличимы. По свойству неотличимых величин отсюда следует неотличимость ξ_0 и ξ_1 .

Неотличимость ξ_2 и ξ_1 следует из того, что они получаются из неотличимых величин $\langle \tilde{u}, y \rangle \sim \langle u, y \rangle$ преобразованием $\langle u, y \rangle \mapsto \langle u, \tilde{v}(u) \rangle$, выполняемым схемой полиномиального размера. \square

Точно таким же рассуждением можно доказать следующее обобщение доказанного утверждения на полиномиальное количество повторений. Пусть $p(n)$ произвольный полином и пусть стратегия F_x разглашает только $f(x)$. Рассмотрим стратегию $F_x^{p(n)}$ заключающуюся в последовательном независимом n -кратном выполнении стратегии F_x .

Лемма 32. *Если стратегия F_x разглашает только $f(x)$, то стратегия $F_x^{p(n)}$ разглашает только $f(x)$.*

Доказательство. Поскольку в рассуждении параметр n меняться не будет, будем опускать его в обозначениях. Зафиксируем симулятор S для стратегии F_x и построим симулятор $S_{p(n)}$ для стратегии $F_x^{p(n)}$. Новый симулятор должен, получив на вход $y = f(x)$ и имея в качестве оракула некоторую неравномерно полиномиальную стратегию G , сгенерировать случайную величину, неотличимую от диалога $c(F_x^{p(n)}, G)$. Заметим, что этот диалог можно определить рекурсивно:

$$\begin{aligned} c(F_x^0, G) &= \langle \rangle, \\ c(F_x^i, G) &= \langle u, c(F_x^{i-1}, G_u) \rangle, \end{aligned}$$

где $u = c(F_x, G)$. Поэтому определим симулятор S_i^G также рекурсивно:

$$\begin{aligned} S_0^G(y) &= \langle \rangle, \\ S_i^G(y) &= \langle \tilde{u}, S_{i-1}^{G_{\tilde{u}}} \rangle, \end{aligned}$$

где $\tilde{u} = S^G(y)$.

Неотличимость $c(F_x^i, G)$ и $S_i^G(y)$ можно доказать по индукции. При $i = 0$ неотличимость очевидна. Индуктивный переход: пусть $c(F_x^i, G)$ и $S_i^G(y)$ неотличимы для любой неравномерно полиномиальной стратегии G . Рассмотрим вспомогательную случайную величину

$$\langle u, S_i^{G_u}(y) \rangle,$$

где $u = c(F_x, G)$. Она неотличима от

$$\langle u, c(F_x^i, G_u) \rangle,$$

поскольку по индуктивному предположению они неотличимы при любом фиксированном u . С другой стороны, она неотличима от случайной величины

$$\langle \tilde{u}, S_i^{G_{\tilde{u}}}(y) \rangle,$$

поскольку получаются применением неравномерно полиномиального отображения $\langle u, y \rangle \mapsto \langle u, S_i^{G_u}(y) \rangle$ к неотличимым величинам $\langle u, y \rangle$ и $\langle \tilde{u}, y \rangle$.

К сожалению, по индукции мы можем доказать неотличимость $c(F_x^i, G)$ и $S_i^G(y)$ только для любого фиксированного i . Чтобы доказать неотличимость полиномиально растущего i , зафиксируем полиномиального размера схему E_n , с вероятностью различения $\varepsilon = 1/\text{poly}(n)$. Затем рассмотрим $p(n) + 1$ гибридных случайных величин вида:

$$\xi_i = \langle u_i, S_{p(n)-i}^{G_{u_i}}(y) \rangle,$$

где $u_i = c(F^i, G)$. При $i = p(n)$ эта случайная величина совпадает с $c(F_x^{p(n)}, G)$, а при $i = 0$ совпадает с $S_{p(n)}^G(y)$. Поэтому найдется i , для которого вероятность различения величин ξ_{i+1} и ξ_i не меньше $\varepsilon/p(n)$. Заметим, что

$$\begin{aligned}\xi_{i+1} &= (u_i, v_i, S_{p(n)-i-1}^{G_{u_i, v_i}}(y)) \\ \xi_i &= (u_i, \tilde{v}_i, S_{p(n)-i-1}^{G_{u_i, \tilde{v}_i}}(y)),\end{aligned}$$

$v_i = c(F, G_{u_i})$ и $\tilde{v}_i = S^{G_{u_i}}(y)$.

Поэтому вероятностная схема $E(u_i, v, S_{p(n)-i-1}^{G_{u_i, v}}(y))$ со входами v, y имеет успех $\varepsilon/p(n)$ в различении случайных величин $\langle v, y \rangle$ и $\langle \tilde{v}, y \rangle$. Можно так зафиксировать u_i (выбрав случайные биты, использованные стратегией F^i), что детерминированная схема $E(u_i, v, S_{p(n)-i-1}^{G_{u_i, v}}(y))$ будет иметь успех в различении $\varepsilon/p(n)$ в различении случайных величин $\langle v, y \rangle$ и $\langle \tilde{v}, y \rangle$, что дискредитирует симулятор S . \square

Задача 38. Пусть стратегия F_x разглашает только $f(x)$, а стратегия G_x — только $g(x)$. Докажите, что стратегия $F_x G_x$, заключающаяся в последовательном выполнении стратегий F_x и G_x , разглашает только $\langle f(x), g(x) \rangle$.

В определении неразглашения требуется, чтобы случайные величина $S^G(f(x))$, была неотличима от случайной величины $c(F_x, G)$ были неотличимы для любого x (точнее, для любой последовательности слов x_n полиномиальной длины). Иногда достаточно более слабого понятия. Пусть задана последовательность распределений вероятностей μ_n на двоичных словах. Скажем, что стратегия F_x разглашает только $f(x)$ относительно μ при наличии общеизвестной информации $g(x)$, если существует вероятностный полиномиальный алгоритм S с оракулом, для которого выполнено следующее. Для любой неравномерно полиномиальной последовательности стратегий $\{G_n\}$ случайная величина $\langle S^G(f(x), g(x)), g(x) \rangle$ вычислительно неотличима от случайной величины $\langle c(F_x, G), g(x) \rangle$ (общеизвестная информация дается и симулятору). В этом определении x выбирается случайно по распределению μ , а не фиксируется, как раньше. Теперь уже важно, что отличитель имеет доступ к $g(x)$, поскольку $g(x)$ не фиксировано.

Если F_x разглашает только $f(x)$, то она разглашает только $f(x)$ относительно любого распределения вероятностей при наличии любой общеизвестной информации. Действительно, допустим некоторая схема E_n

полиномиального размера отличает случайные величины $\langle S^G(f(x), g(x)), g(x) \rangle$ и $\langle c(F_x, G), g(x) \rangle$. Тогда, что для бесконечно многих n среднее по μ значение разности вероятностей

$$\Pr[E(S^G(f(x), g(x)), g(x)) = 1] - \Pr[E(c(F_x, G), g(x)) = 1]$$

не меньше $1/\text{poly}(n)$. Тогда для каждого из этих n найдется x_n , для которого разность не меньше $1/\text{poly}(n)$. Запаяв $g(x_n)$ в E , мы получим отличитель $S^G(f(x_n))$ и $c(F_{x_n}, G)$.

Обратное неверно. Пусть, например, μ_n равномерно распределено среди всех слов длины n , а F_x сообщает x и после этого останавливает общение. Тогда F_x имеет нулевое разглашение относительно μ_n при отсутствии общеизвестной информации: симулятор выдает на выход равномерно распределенную последовательность длины n . При этом очевидно, что F_x не имеет нулевого разглашения.

Более того, при отсутствии общеизвестной информации (то есть, при $g(x) \equiv$ (пустое слово)) любая полиномиальная вероятностная стратегия F_x имеет нулевое разглашение относительно любого полиномиально генерируемого распределения вероятностей. Симулятор выбирает x случайным образом относительно этого распределения и моделирует общение F_x и данной ему стратегии G . Поэтому нулевое разглашение относительно μ не монотонно по $g(x)$: при возрастании общедоступной информации стратегия с нулевым разглашением может перестать быть таковой. Например, нетрудно построить полиномиальную стратегию S_x и полиномиально вычислимую функцию g такие, что S_x не имеет нулевого разглашения относительно равномерного распределения на словах длины n при общеизвестном $g(x)$.

Задача 39. Приведите пример таких S_x и g .

Недостатком понятие неразглашения относительно данного распределения является тот факт, что для него неверна Лемма 32 (при условии существования односторонних функций). Это следует из следующих двух задач.

Задача 40. Пусть $h_n : \{0, 1\}^n \rightarrow \{0, 1\}$ является трудным битом для функции $g_n : \{0, 1\}^n \rightarrow \{0, 1\}^n$ (определение на стр. 20). Рассмотрим стратегию F_x , которая сообщает пару $\langle g(x), h(x) \rangle$ и после этого останавливается. Докажите, что F_x имеет нулевое разглашение относительно равномерного распределения на $\{0, 1\}^n$ при общеизвестном $g(x)$.

Задача 41. Пусть $g_n : \{0, 1\}^n \rightarrow \{0, 1\}^n$ необратимая функция (с равномерным распределением на входах). Пусть F_x — стратегия, которая выбирает случайно строку y длины n , сообщает пару $\langle y, y \odot x \rangle$ и останавливается. Докажите, что F_x имеет нулевое разглашение относительно равномерного распределения на $\{0, 1\}^n$ при общеизвестном $g(x)$. Докажите, что стратегия F_x^{2n} не имеет нулевого разглашения относительно того же распределения при общеизвестном $g(x)$.

9.3 Интерактивные доказательства с нулевым разглашением

Определим теперь понятие интерактивного доказательства с нулевым разглашением. Пусть имеется интерактивный алгоритм V и вероятностная стратегия P_x с параметром x . Такая пара называется протоколом интерактивного доказательства с нулевым разглашением для множества слов L , если выполнены три условия.

- (1) Для всех $x \in L$ выполнено $\Pr[V^{P_x}(x) = 1] \approx 1$.
- (2) Для всех $x \notin L$ для любой стратегии F выполнено $\Pr[V^F(x) = 1] \approx 0$.

Приблизительные равенства в обоих условиях при стремлении длины x к бесконечности. Эти условия гарантируют, то что L принадлежит классу IP.

- (3) Стратегия P_x разглашает только $\langle x, L(x) \rangle$.

Множество всех языков, для которых существует протокол интерактивного доказательства с нулевым разглашением называется ZKP (Zero Knowledge Proofs).

Теорема 33. Язык GI, состоящий из множества всех пар $\langle G_0, G_1 \rangle$ изоморфных графов, принадлежит ZKP.

Доказательство. Игру начинает Доказывающий. Если графы G_0, G_1 неизоморфны, то он останавливает игру. Иначе он фиксирует некоторый изоморфизм τ этих графов: $\tau G_1 = G_0$. Затем он посылает Проверяющему граф $H = \pi G_0$, где π случайную перестановку множества $\{1, \dots, n\}$ вершин графа G_0 . Получив граф H , проверяющий посылает Доказывающему случайно выбранный бит α . Затем Доказывающий оставляет игру, если ему прислали не бит (то есть длина присланного сообщения не равна 1), а иначе посылает перестановку $\rho = \pi \circ \tau^\alpha$. Эта перестановка устанавливает изоморфизм графов G_α и H , то есть, $\pi \circ \tau^\alpha(G_\alpha) = \pi(G_0) = H$.

Проверяющий проверяет, в самом ли деле $\rho(G_\alpha) = H$ и останавливает игру досрочно, если это не так. Если игра не остановлена, то все продолжается сначала, с новыми π, α (изоморфизм τ не меняется). После n повторений Проверяющий останавливает игру, и, если игра не была остановлена досрочно, выдает 1, а иначе 0.

Условие (1) очевидно выполнено. Проверим условие (2). Пусть Доказывающий применяет некоторую стратегию F (возможно отличную от стратегии P). Если графы G_0, G_1 неизоморфны, то посланный им в i -ом раунде граф H не изоморфен хотя бы одному из графов G_0, G_1 . С вероятностью не менее $1/2$ Доказывающий укажет в качестве α номер того графа, которому H не изоморфен. Если это произошло, то какую бы перестановку ρ ни прислал Доказывающий, равенство $\rho(G_\alpha) = H$ не выполнено, и Проверяющий досрочно остановит игру. Вероятность этого события не менее одной второй при условии любых ходов в предыдущих раундах, поэтому вероятность досрочной остановки игры хотя бы в одном раунде не менее $1 - 2^{-n}$.

Осталось проверить условие (3). Поскольку Доказывающий применяет одну и ту же стратегию во всех раундах, по лемме 32 достаточно доказать, что в первом раунде разглашается только $\langle x, L(x) \rangle$.

Симулятор действует так. Пусть F некоторая (детерминированная) стратегия Проверяющего. Будем сначала предполагать, что нам заранее известно, что F на любом графе выдает 0 или 1. Если $L(x) = 0$, то генерация записи общения тривиальна, потому что игра заканчивается, не начавшись. Пусть $L(x) = 1$, то есть графы изоморфны. Тогда нам нужно по G_0, G_1 сгенерировать случайной величиной, вычислительно неотличимую от величины $\xi = \langle \pi(G_0), \alpha, \pi \circ \tau^\alpha \rangle$. При этом мы не знаем τ (иначе генерация была бы тривиальной — мы могли бы просто запустить стратегии Проверяющего и Доказывающего).

Нетрудно понять, что распределение ξ есть равномерное распределение на тройках $\langle H, \alpha, \rho \rangle$ таких, что $\rho(G_\alpha) = H$ и $F(H) = \alpha$ (будем называть такие тройки удачными). Действительно, для каждой перестановки π имеется удачная тройка $\langle H, \alpha, \rho \rangle$ — та, которая получена, если на первом шаге Доказывающий выбрал перестановку π . С другой стороны, каждая удачная тройка H, α, ρ появится в беседе, если на первом шаге Доказывающий выберет перестановку $\pi = \rho \circ \tau^{-\alpha}$. Таким образом, удачных троек ровно $n!$ и все они равновероятны.

Теперь можно запустить хорошо известный алгоритм генерации равномерного распределения в данном множестве. Сначала заметим, что

первая компонента любой удачной тройки однозначно определяется второй и третьей компонентами. Поэтому можно взять случайную пару α, ρ , найти $H = \rho(G_\alpha)$ и проверить будет ли тройка $\langle H, \alpha, \rho \rangle$ удачной (то есть, $F(H) = \alpha$). Если нет, то повторить попытку, взяв новую пару α, ρ . Чтобы этот алгоритм за полиномиальное от n время находил с большой вероятностью удачную тройку, нужно чтобы удачных троек было достаточно много. Мы утверждаем, что вероятность того, что тройка $\langle \rho(G_\alpha), \alpha, \rho \rangle$ удачна, равна $1/2$. Действительно вероятность того, что $F(\rho(G_\alpha)) = \alpha$ равна среднему арифметическому вероятностей событий $F(\rho(G_0)) = 0$ и $F(\rho(G_1)) = 1$. Поскольку графы G_0 и G_1 изоморфны, случайные величины $\rho(G_0)$ и $\rho(G_1)$ имеют одинаковое распределение, поэтому вероятность события $F(\rho(G_0)) = 0$ равна вероятности события $F(\rho(G_1)) = 0$. Значит вероятность найти успешную тройку в одной попытке равна среднему арифметическому вероятностей событий $F(\rho(G_1)) = 0$ и $F(\rho(G_1)) = 1$. Поскольку эти события взаимно дополнительные (здесь мы используем, что стратегия F выдает всегда бит), эта вероятность равна $1/2$. Таким образом, после n попыток вероятность неудачи будет равна 2^{-n} и в этом случае мы можем выдать любую тройку.

Как генерировать запись сообщений в общем случае. Обозначим через \hat{F} стратегию, которая в том случае, когда F выдает не бит, выдает, скажем ноль, а в остальных случаях работает так же, как F . Симулятор действует как описано, используя \hat{F} вместо F . После получения тройки $\langle H, \alpha, \rho \rangle$ он применяет F к графу H и, если получится не бит, то заменяет в этой тройке α (которое в этом случае равно 0) на $F(H)$, а ρ на пустое слово. Это преобразование переводит распределение $c(P_x, \hat{F})$ в распределение $c(P_x, F)$ (поскольку при каждом фиксированном π , тройка полученная, в игре стратегий P_x и F получается этим преобразованием из тройки полученной в игре стратегий P_x и \hat{F}). Поэтому это преобразование переводит и любое распределение, статистически неотличимое от распределения $c(P_x, \hat{F})$ в распределение, статистически неотличимое от распределения $c(P_x, F)$. \square

9.4 Интерактивный протокол для любого языка из NP

Ясно, что алгоритм привязки к биту можно использовать дважды, привязываясь к паре битов. В следующей протоколе мы будем кодировать

каждый из трех элементов множества $\{1, 2, 3\}$ парой битов и с помощью привязки к паре битов, будем осуществлять привязку к цвету, то есть к одному из чисел 1,2,3. При этом, если исходный протокол удовлетворял условию (2') (определение на стр.59), то полученный протокол (S, R, T) привязки к цвету также удовлетворяет аналогичному условию. А поэтому (смотри задачу 33 на стр. 59) существует функция $K(c)$ со значениями 1,2,3 такая, что для любой неравномерно полиномиальной последовательности стратегий $\{S_n^*\}$ с приблизительно единичной вероятностью для всех k выполнено $T(c(S^*, R), k) \in \{K(c(S^*, R)), \perp_S\}$.

Теорема 34. *Если существует протокол привязки к биту, удовлетворяющий усиленному требованию (2') (определение на стр.59), то существует интерактивный протокол с нулевым разглашением для множества графов, раскрашиваемых в три цвета.*

Доказательство. Сначала построим интерактивный протокол (P, V) с такими свойствами.

(1) Если граф $G = (\{1, \dots, n\}, E)$ можно раскрасить в три цвета, то с вероятностью примерно 1 (при n стремящемся к бесконечности) $V(G)^{P(G)} = 1$.

(2) Если граф $G = (\{1, \dots, n\}, E)$ нельзя раскрасить в три цвета, то для любой стратегии P^* с вероятностью не меньше $1/|E|$ минус ничтожно малая величина (при n стремящемся к бесконечности) выполнено $V(G)^{P^*} = 0$.

(3) Алгоритм $P(G)$ разглашает только G и то, можно ли граф G раскрасить в три цвета.

Интерактивный протокол (P, V) состоит в следующем. Если граф нельзя раскрасить в три цвета, то алгоритм P сразу останавливает общение. Иначе он выбирает корректную 3-раскраску ϕ исходного графа $G = (\{1, \dots, n\}, E)$. Затем алгоритм P выбирает случайную перестановку π множества цветов $\{1, 2, 3\}$ и применяет ее к данной раскраске ϕ . Полученную раскраску будем обозначать через α . Затем Доказывающий и Проверяющий последовательно для каждой вершины $u \in \{1, \dots, n\}$ запускают алгоритмы S, R привязки к цвету $\alpha(u)$ каждой вершины $u \in \{1, \dots, n\}$. То есть, происходит беседа интерактивных алгоритмов $S_\alpha = S_{\alpha(1)} \dots S_{\alpha(n)}$ и R^n . В качестве параметра безопасности берется n . Обозначим через $k_v, v \in \{1, \dots, n\}$, ключи, выданные алгоритмом S_α .

Затем алгоритм V выбирает случайным образом некоторое ребро (u, v) в графе и посылает Доказывающему пару (u, v) . В ответ алгоритм

P посылает k_u, k_v . Алгоритм V , применяя алгоритм T раскрытия сундучка, проверяет что цвета $\alpha(u)$ и $\alpha(v)$ различны. Если они оказались совпадающими или алгоритм T сообщил \perp_S хотя бы в одном из двух случаев, то алгоритм V выдает 0, а иначе 1.

Проверим требования к интерактивному протоколу с нулевым разглашением. Требование (1) очевидно выполнено: если граф можно раскрасить в три цвета, то P убедит V с вероятностью примерно 1.

Проверим требование (2). По условию существует функция $K(c)$ со значениями 1,2,3 такая, что для любой стратегии S^* с приблизительно единичной вероятностью для всех k выполнено

$$T(c(S^*, R), k) \in \{K(c(S^*, R)), \perp_S\}. \quad (3)$$

Пусть граф G нельзя раскрасить в 3 цвета. Обозначим через $c_u(P^*, R)$ протокол общения на стадии привязки к цвету вершины u . Применим к $c_u(P^*, R)$ функцию K и рассмотрим полученный цвет в качестве цвета вершины u . Поскольку полученная раскраска не корректна, найдется ребро (u, v) , для которого $K(c_u(P^*, R)) = K(c_v(P^*, R))$. С вероятностью не менее $1/|E|$ Проверяющий выберет именно это ребро. Теперь применим условие (3) к следующей стратегии S^* : моделируем общение стратегии P^* и алгоритма R^{w-1} (последовательное выполнение $w - 1$ раз алгоритма R), а затем продолжаем применять стратегию P^* для привязки к цвету вершины w , общаясь со стратегией R . Для этой стратегии $c_w(S^*, R) = c_w(P^*, R)$, поэтому для любой вершины w примерно единичной вероятностью для любого ключа k выполнено

$$T(c_w(P^*, R), k) \in \{K(c_w(P^*, R)), \perp_S\}.$$

В частности это верно для вершин u, v и ключей k_u, k_v , предоставленных стратегией P^* . А значит с вероятностью не меньше $1/|E|$ (минус пренебрежимо малая величина) Проверяющий выдаст 0.

Осталось проверить требование (3). Пусть Проверяющий вместо предписанной стратегии применяет некоторую последовательность схем V^* полиномиального размера. Будем обозначать через $c(P, V^*)$ запись общения P и V^* . Если граф G не 3-раскрашиваем, то $c(P, V^*)$ состоит из пустого слова и может быть порождено по известной нам информации, что граф не 3-раскрашиваем. Пусть граф является 3-раскрашиваемым. Нам нужно придумать алгоритм с оракулом V^* порождающий случайную величину, вычислительно неотличимую от случайной величины $c(P, V^*)$.

Пусть β произвольная, возможно некорректная, раскраска графа. Рассмотрим следующую стратегию доказывающего, которую мы обозначим через P_β : запускаем алгоритмы S_β и V^* , и по получении ребра (u, v) выдаём пару ключей k_u, k_v . Таким образом, исходная стратегия Доказывающего P заключается в выполнении P_α для случайной корректной раскраски α вида $\pi\phi$.

Алгоритм порождения случайной величины, неотличимой от $c(P, V^*)$, таков. Выберем случайную (возможно некорректную) раскраску β и смоделируем общение P_β с V^* . Скажем, что нам повезло, если V^* после стадии привязки выдаст ребро (u, v) , для которого $\beta(u) \neq \beta(v)$. В этом случае выдаём полученную запись беседы в качестве результата. Иначе повторяем то же самое, используя новую случайную раскраску β . И так действуем до тех пор, пока нам не повезет или не произойдет n неудачных попыток.

Докажем, что вероятность везения в каждой попытке примерно равна $2/3$, поэтому вероятность того, что во всех n попытках не повезло, пренебрежимо мала. Зафиксируем ребро (u, v) и два различных цвета a, b . Вероятность везения равна сумме по всем (u, v) и a, b вероятности того, что $V^*(c(S_\beta, V^*)) = (u, v)$ (то есть, V^* , изучив запись своего общения с S_β , выдаёт (u, v)) и при этом $\beta(u) = a, \beta(v) = b$. Поскольку мы зафиксировали цвета вершин u и v , при подсчете этой вероятности стратегия S_β может быть заменена на последовательную композицию стратегий $S_{\beta(1)}, \dots, S_a, \dots, S_b, \dots, S_{\beta(n)}$. Будем эту композицию обозначать через $S_{\beta abuv}$. Поскольку события $V^*(c(S_{\beta abuv}, V^*)) = (u, v)$ и $\beta(u) = a, \beta(v) = b$ независимы, вероятность их пересечения равна произведению их вероятностей, то есть, одной девятой вероятности события $V^*(c(S_{\beta abuv}, V^*)) = (u, v)$. Теперь воспользуемся тем, что каждая из стратегий $S_{\beta(1)}, \dots, S_a, \dots, S_b, \dots, S_{\beta(n)}$ имеет нулевое разглашение, а значит и их композиция тоже. Значит, существует полиномиальный алгоритм с оракулом V^* , генерирующий случайную величину θ , вычислительно неотличимую от протокола общения $S_{\beta abuv}$ и V^* . Поэтому вероятность того, что $V^*(S_{\beta abuv}) = (u, v)$, примерно равна вероятности того, что $V^*(\theta) = (u, v)$. Отсюда следует, что вероятность везения примерно равна одной девятой суммы по ребрам (u, v) и парам a, b разных цветов вероятности того, что $V^*(\theta) = (u, v)$. Действительно, в этой сумме полиномиальное от n количество слагаемых, поэтому от изменения каждого слагаемого на пренебрежимо малую величину сумма изменится незначительно. Сумма по всем (u, v) вероятности того, что $V^*(\theta) = (u, v)$, равна 1. Поскольку существует ровно 6

пар различных цветов a, b , мы в результате получаем примерно $2/3$.

Итак, наш алгоритм порождает случайную величину, статистически неотличимую от следующей случайной величины η . Вероятность того, что η принимает значение c , есть вероятность того, что $c = c(P_\beta, V^*)$ при условии, что $V^*(c(S_\beta, V^*))$ равно такому ребру (u, v) , для которого $\beta(u) \neq \beta(v)$.

Пусть A произвольная схема полиномиального от n размера. Нам надо доказать, что вероятность события

$$A(c(P_\alpha, V^*)) = 1$$

примерно равна трём вторым вероятности события

$$A(c(P_\beta, V^*)) = 1, \beta(u) \neq \beta(v).$$

Представим каждое из этих событий как объединение непересекающихся событий (по всем ребрам (u, v) и всем парам различных цветов a, b):

$$A(c(P_{\alpha abuv}, V^*)) = 1, V^*(c(S_{\alpha abuv}, V^*)) = (u, v), \alpha(u) = a, \alpha(v) = b \quad (4)$$

и

$$A(c(P_{\beta abuv}, V^*)) = 1, V^*(c(S_{\beta abuv}, V^*)) = (u, v), \beta(u) = a, \beta(v) = b. \quad (5)$$

Через $P_{\beta abuv}$ мы обозначили стратегию, заключающуюся в последовательном выполнении стратегий $S_{\beta(1)}, \dots, S_a, \dots, S_b, \dots, S_{\beta(n)}$ и последующей выдаче пары ключей. Вероятность в этих событиях считается по случайному выбору α, β и внутренним случайным датчикам стратегий.

Нам достаточно установить, что вероятность события (4) примерно равна трём вторым вероятности события (5). Фиксируем u, v, a, b . Событие (4) есть пересечение четырех событий, причем последние два события не зависят от первых двух. Вероятность пересечения последних двух событий равна одной шестой, поэтому вероятность события (4) равна одной шестой вероятности события

$$A(c(P_{\alpha abuv}, V^*)) = 1, V^*(c(S_{\alpha abuv}, V^*)) = (u, v). \quad (6)$$

Аналогичным образом, вероятность события (5) равна одной девятой вероятности события

$$A(c(P_{\beta abuv}, V^*)) = 1, V^*(c(S_{\beta abuv}, V^*)) = (u, v). \quad (7)$$

Следовательно, нам достаточно доказать, что вероятности событий (6) и (7) примерно равны. Для этого докажем, что они обе примерно равны вероятности некоторого третьего события.

Сначала займёмся первым из двух событий. Будем использовать то, что стратегия S_c имеет нулевое разглашение. В частности, все стратегии $S_{\alpha(1)}, \dots, S_a, \dots, S_b, \dots, S_{\alpha(n)}$ имеют нулевое разглашение. Однако, стратегия $P_{\alpha abuv}$ уже не имеет нулевого разглашения, поскольку она разглашает пару ключей. Оценим сверху разглашение информации стратегией $P_{\alpha abuv}$ следующим образом. Все стратегии $S_{\alpha(i)}$ при $i \neq a, b$ имеют нулевое разглашение. Стратегию S_a будем рассматривать как детерминированную стратегию со случайным входом r и обычным входом a , и обозначать через S_{ar} . Очевидно, что S_{ar} разглашает только a и r . Аналогично поступим и со стратегией S_b и будем обозначать полученную стратегию через S_{bs} . Будем рассматривать стратегию $P_{\alpha abuv}$ как стратегию, которая сначала выбирает случайным образом r, s затем последовательно выполняет следующую стратегию, обозначаемую через $P_{\alpha abuvrs}$: последовательно выполнить $S_{\alpha(1)}, \dots, S_{ar}, \dots, S_{bs}, \dots, S_{\alpha(n)}$, а затем выполнить стратегию, которая получает ребро и в ответ посылает пару ключей. Будем считать, что независимо от ребра, присланного доказывающим, последняя стратегия посылает ключи $k_u = k(r, a)$ и $k_v = k(s, b)$, где (u, v) — зафиксированное ребро. Ясно, что вероятность события (6) от такой подмены не изменится. (Напомним, что без ограничения общности мы можем считать, что ключ, выдаваемый стратегией S_c зависит только от s её случайных битов, задача 34 на стр. 60.) К стратегии $P_{\alpha abuvrs}$ можно применить лемму о последовательном выполнении. Поэтому стратегия $P_{\alpha abuvrs}$ разглашает только a, b, u, v, r, s . Таким образом, по a, b, u, v, r, s мы можем сгенерировать случайную величину, которая вычислительно неотличима от диалога стратегий $P_{\alpha abuvrs}$ и V^* . А следовательно, по a, b, u, v можем породить случайную величину, вычислительно неотличимую от диалога стратегий $P_{\alpha abuv}$ и V^* . Будем эту случайную величину обозначать через θ_{abuv} , а ее начало, имитирующее диалог $S_{\alpha abuv}$ и V^* , через ξ_{abuv} . Итак, для любых a, b и u, v вероятность события (6) примерно равна вероятности события

$$A(\theta_{abuv}) = 1, \quad V^*(\xi_{abuv}) = (u, v).$$

Заметим, что в этом рассуждении мы никак не использовали свойств распределения вероятностей, по которому выбирается α . Поэтому то же самое верно для вероятности события (7).

Итак, мы построили протокол, удовлетворяющий свойствам (1), (2) и (3). Теперь понизим вероятность ошибки с помощью последовательного повторения этого протокола $n|E|$ раз. Новый алгоритм Проверяющего выдает 1, если в каждом из $n|E|$ повторений алгоритм V выдал 1. В силу леммы о последовательном выполнении, стратегия Доказывающего разглашает только G и раскрашиваемость графа. Вероятность ошибки станет пренебрежимо малой в силу свойств последовательного повторения. Свойство (1) очевидно сохранится. \square

10 Протоколы аутентификации

Протоколом аутентификации называется последовательность доступных случайных величин $\langle d_n, e_n \rangle$ (закрытый и открытый ключи) и пара полиномиальных вероятностных интерактивных алгоритмов P, V , алгоритм Доказывающего и алгоритм проверки. Оба алгоритма получают на вход параметр безопасности n и работают время, ограниченное полиномом от n . Алгоритм P получает на вход, кроме параметра безопасности, ключ d , алгоритм V — ключ e .

Требования таковы:

(1) С вероятностью, приблизительно равной 1, выполнено $V^{P_d}(e) = 1$. Вероятность берется по распределению на паре ключей (e, d) и по исходам случайных бросаний, выполняемых P и V .

(2) Для любого полинома $p(n)$ алгоритм $P_d^{p(n)}$ имеет нулевое разглашение при общеизвестном e относительно распределения, генерируемого алгоритмом K . (Для простоты обозначений мы предполагаем, что e есть функция от d .)

(3) Для любой неравномерной стратегии F_n вероятность события $V^{F_n(e)}(e) = 1$ пренебрежимо мала. Вероятность берется по случайному выбору e и по исходам случайных бросаний, выполняемых V .

Условия (2) и (3) гарантируют следующее. Пусть противник изготовил фальшивый банкомат, который вместо правильной стратегии V использует некоторую неравномерно полиномиальную стратегию V^* . Пусть владелец карточки $p(n)$ раз прошел аутентификацию в этом фальшивом банкомате. Тогда противнику становится известным $u = c(P_d^{p(n)}, V^*(e))$ (запись беседы $P_d^{p(n)}$ и $V^*(e)$). После этого противник, используя некоторую неравномерно полиномиальную стратегию P^* , которая имеет на входе u, e , хочет сам пройти аутентификацию в настоящем банкомате.

Из условий (2) и (3) следует, что шансы этого пренебрежимо малы (при случайном выборе d, e). Действительно, по условию (2) у нас имеется симулятор S . Применим его к e с оракулом $V^*(e)$. Он выдаст случайную величину $\tilde{u} = S^{V^*(e)}(e)$ такую, что случайные величины $\langle u, e \rangle$ и $\langle \tilde{u}, e \rangle$ вычислительно неотличимы. Рассмотрим теперь следующую попытку их отличить. Получив на вход $\langle v, e \rangle$, запускаем беседу стратегий $V(e)$ и $P^*(v, e)$ и выдаем в качестве результата ответ V . Этот отличитель задается схемой полиномиального размера. Вероятность того, что он выдаст 1 на входе $\langle \tilde{u}, e \rangle$ пренебрежимо мала по свойству (3). Из неотличимости $\langle u, e \rangle$ и $\langle \tilde{u}, e \rangle$ следует, что и вероятность того, что он на входе $\langle u, e \rangle$ (где $u = c(P_d^{p(n)}, V^*(e))$) выдаст 1, также пренебрежимо мала.

Теорема 35. *Если функция Рабина необратима, то существует протокол аутентификации.*

Доказательство. Пусть $\langle m_n, x_n \rangle$ трудная случайная величина для функции Рабина. Напомним, что m_n, x_n — n -битовые натуральные числа, а функция Рабина f_n определена как $f_n(m, x) = \langle x^2 \bmod m, m \rangle$. Закрытый ключ d_n равен паре $\langle m_n, x_n \rangle$ а открытый ключ e_n равен паре $\langle m_n, x_n^2 \bmod m_n \rangle$.

Алгоритмы P, V работают так.

- 1) Алгоритм P выбирает случайный обратимый вычет y и посылает $z = y^2 \bmod m$ (с вероятностью пренебрежимо малой он не сможет найти такого вычета, в этом случае он останавливает беседу).
- 2) Алгоритм V посылает случайный бит α .
- 3) Алгоритм P проверяет, является ли присланное ему одним битом (если это не так, то в качестве α он берет первый бит полученного слова) и посылает $u = yx^\alpha$. Заметим, что $u^2 = zx^{2\alpha} \pmod{m}$.
- 4) Алгоритм V проверяет равенство $u^2 = zx^{2\alpha} \pmod{m}$ (он может это сделать, поскольку знает x^2, m) и проверяет, взаимно просты ли u и m . Если что либо из этого не выполнено, он останавливает беседу досрочно и выдает 0. Иначе повторяются 1)–4), причем каждая из сторон использует новые случайные биты. Если после n повторений общение не остановлено досрочно, алгоритм V останавливает беседу и выдает 1.

Условие (1) очевидно выполнено. Условие (2) проверяется следующим образом. По лемме 32 нам достаточно доказать, что в ходе однократного выполнения пунктов 1)–3) разглашается только m и x^2 . Пусть Проверяющий выполняет какую-то неравномерно полиномиальную стратегию F , которая по квадратичному вычету z сообщает бит $F(z)$ (если $F(z)$

состоит более чем из одного бита, то в дальнейших рассуждениях надо заменить $F(z)$ на $F'(z)$ равное первому биту $F(z)$). Нам надо, используя F , как оракул, сгенерировать случайную величину, вычислительно неотличимую от случайной величины

$$c(P(x, m), F) = \langle y^2, F(y^2), yx^{F(y^2)} \rangle.$$

Здесь y выбирается по распределению, статистически неотличимому от равномерного распределения среди всех обратимых вычетов по модулю m . Если мы заменим распределение y на равномерное, то новое распределение на тройках $\langle y^2, F(y^2), yx^{F(y^2)} \rangle$ будет статистически неотличимо от старого, поэтому достаточно сгенерировать случайную величину, вычислительно неотличимую от случайной величины

$$\xi = \langle y^2, F(y^2), yx^{F(y^2)} \rangle$$

при равномерном распределении на y . На самом деле, мы сгенерируем величину, даже статистически неотличимую от ξ .

Назовем тройку $\langle z, \alpha, u \rangle$ удачной, если $F(z) = \alpha$, $u^2 = zx^{2\alpha}$ и u взаимно просто с m . Очевидно, что все тройки, появляющиеся в беседе, удачны. С другой стороны, никаких других удачных троек нет: нетрудно проверить, что любая удачная тройка $\langle z, \alpha, u \rangle$ появится в беседе, если на первом шаге Доказывающий выберет $y = ux^{-\alpha}$. Кроме того, для каждой удачной тройки $\langle z, \alpha, u \rangle$ такое y единственно. Поэтому случайная величина ξ равномерно распределена на множестве всех удачных троек.

Следовательно, по x^2, m нам нужно сгенерировать распределение, статистически неотличимое от равномерного распределения на множестве удачных троек. Заметим, что в удачной тройке первая компонента однозначна определяется второй и третьей компонентой: $z = u^2x^{-2\alpha}$. Значит, если выбирать случайно пару α, u и выдавать тройку $\langle u^2x^{-2\alpha}, \alpha, u \rangle$, если она оказалось удачной (то есть, $F(z) = \alpha$ и $(u, m) = 1$), то все удачные тройки будут иметь равные шансы быть выданными.

Какова вероятность в одной попытке найти удачную тройку? Пусть u выбирается равномерно среди всех вычетов, взаимно простых с m . Тогда вероятность того, что тройка $\langle u^2x^{-2\alpha}, \alpha, u \rangle$ удачна равна

$$\Pr[V(u^2) = 0]/2 + \Pr[V(u^2x^{-2}) = 1]/2.$$

Второе слагаемое можно заменить на слагаемое $\Pr[V(u^2) = 1]/2$, так как u^2 и u^2x^{-2} имеют одинаковое распределение. Поэтому наша вероятность

равна

$$\Pr[V(u^2) = 0]/2 + \Pr[V(u^2) = 1]/2 = 1/2.$$

При замене равномерного распределения среди всех вычетов, взаимно простых с m , на статистически неотличимое от него равномерное распределение вероятность изменится незначительно, а значит, будет приблизительно равна $1/2$.

Повторяя n раз попытку найти удачную тройку, мы преуспеем хотя бы в одной попытке с вероятностью, примерно равной 1 . В случае неудачи выдадим любую тройку.

Осталось проверить условие (3). Пусть для некоторой неравномерно полиномиальной стратегии F вероятность события $V^{F(m,x^2)}(m, x^2) = 1$ не пренебрежимо мала. Напомним, что вероятность берется по случайному выбору m, x по распределению трудному для функции Рабина. То есть, нам известно, что не существует последовательности схем полиномиального размера D_n , которая по этому распределению для бесконечно многих n по паре m, x^2 выдает некоторый корень из x^2 с вероятностью не менее $1/\text{poly}(n)$. Чтобы получить противоречие, определим такую последовательность D_n .

Нам известно, что для бесконечно многих n вероятность события $V^{F(m,x^2)}(m, x^2) = 1$ больше $\varepsilon = 1/\text{poly}(n)$. Фиксируем одно из таких n . Сначала рассмотрим совсем простой случай: вероятность события $V^{F(m,x^2)}(m, x^2) = 1$ в точности равна 1 . Стратегия F для любых данных m, x^2 сначала посылает Проверяющему некоторый вычет $z = z(m, x^2)$. Затем в зависимости от присланного ему α она присылает некоторый вычет. Будем обозначать вычет, присланный в ответ на $\alpha = 0$ через $u_0 = u_0(m, x^2)$, а вычет, присланный в ответ на $\alpha = 1$, через $u_1 = u_1(m, x^2)$. По предположению с вероятностью 1 выполнено $u_0^2 = z$, $u_1^2 = zx^2$ и u_0, u_1 взаимно просты с m , а следовательно, $x^2 = (u_1/u_0)^2$. Таким образом, следующий алгоритм обращает функцию Рабина с единичной вероятностью:

Для данных m, x^2 применяем F к паре (m, x^2) . Обозначим через z полученный вычет. Применяем F к обоим тройкам $((m, x^2), z, 0)$ и $((m, x^2), z, 1)$. Мы получим два вычета u_0, u_1 . Выдаем в качестве ответа их частное.

Заметим, что реальный Доказывающий отказался бы в одном раунде отвечать на два разных α . Когда мы обращаем функцию Рабина, мы

используем схему, реализующую его стратегию, и можем делать это так, как запрещено в процессе аутентификации.

Теперь рассмотрим чуть более сложный случай: пусть известно, что вероятность события $V^{F(m,x^2)}(m, x^2) = 1$ не меньше $3/4$. В этом случае опять достаточно использовать, что с вероятностью не менее $3/4$ беседа не будет остановлена в первом раунде. Среднее арифметическое вероятностей событий “ $u_0^2 = z$ и u_0 взаимно просто с m ” и “ $u_1^2 = zx^2$ и u_1 взаимно просто с m ” не меньше $3/4$. Значит сумма их вероятностей не меньше $3/2$, поэтому вероятность их пересечения не меньше $1/2$. Таким образом, с вероятностью не меньше $1/2$ будет выполнено $x^2 = (u_1/u_0)^2$. Тот же самый алгоритм будет теперь обращать функцию Рабина с вероятностью не менее $1/2$.

Теперь рассмотрим общий случай. Обозначим через p_1 вероятность того, что беседа не остановлена в первом раунде, а через p_i — вероятность того, что беседа не остановлена в i -ом раунде при условии, что она не остановлена раньше. Нам дано, что $p_1 p_2 \cdots p_n \geq \varepsilon = 1/\text{poly}(n)$. Значит, существует такое i , для которого $p_i \geq (1/\text{poly}(n))^{1/n}$. Последовательность $(1/\text{poly}(n))^{1/n}$ стремится к 1, поэтому при достаточно больших n выполнено неравенство $p_i > 3/4$. Зафиксируем любое i , для которого это выполнено, и рассмотрим следующую вероятностную процедуру обращения функции Рабина.

Для данных m, x^2 , применяя F к подходящим входам, находим последовательность сообщений c , которая появится в беседе в первых $i - 1$ раундах. При этом $\alpha_1, \dots, \alpha_{i-1}$ (сообщения, посылаемые Проверяющим) выбираем случайным образом. Находим $z = F(c)$, и $u_0 = F(c, z, 0)$ и $u_1 = F(c, z, 1)$. Выдаем в качестве ответа частное u_1/u_0 .

Нам известно, что вероятность того, что u_0, u_1 обратимы и квадрат их частного равен x^2 , при условии, что беседа дойдет до i -ого раунда, не меньше $1/2$. Поскольку вероятность самого условия не меньше ε , безусловная вероятность того, что мы найдем корень из x^2 , не меньше $\varepsilon/2$.

В приведенном рассуждении есть неточность — мы не знаем, чему равно i . Ее легко исправить: например, можно пробовать все i . А именно, для каждого i применять F к $\alpha = 0$ и к $\alpha = 1$. Если нам не повезло и корень из x^2 не найден, то выбираем α случайно и переходим к следующему раунду.

А можно рассуждать и так: поскольку обратитель является схемой полиномиального размера, которая не обязана вычисляться эффективно по n , можно запаковать i в схему-обратитель. \square

11 Протоколы электронной подписи

11.1 Подпись одного бита

Протоколом подписи одного бита называется последовательность доступных случайных величин $\langle d_n, e_n \rangle$ (закрытый и открытый ключи) и пара детерминированных (неинтерактивных) алгоритмов S, V , называемыми алгоритмом подписи и алгоритм проверки, соответственно. Оба алгоритма получают на вход параметр безопасности n и работают время, ограниченное полиномом от n . Алгоритм P получает на вход, кроме параметра безопасности, ключ d и один бит σ и выдает подпись s . Алгоритм V получает на вход, кроме параметра безопасности, ключ e , бит σ и подпись s и выдает 0 или 1 (отвергает или принимает подпись). Требования таковы:

(1) С вероятностью, приблизительно равной 1, для всех $\sigma = 0, 1$ выполнено $V(e, \sigma, S(d, \sigma)) = 1$ (параметр n мы опускаем). Вероятность берется по распределению на паре ключей (e, d) .

(2) Мы требуем защищенности протокола от следующей атаки. Противник, изучив открытый ключ e , просит Подписывающего подписать некоторый бит σ . Подписывающий (не подозревая подвоха) дает ему требуемую подпись $s = S(d, \sigma)$. Противник изучает s и выдает поддельную подпись s' . Атака считается успешной, если s' признается правильной подписью под противоположным битом $\bar{\sigma}$, то есть, $V(e, \bar{\sigma}, s') = 1$. Итак, мы требуем, чтобы для любой последовательности схем C_n, D_n полиномиального от n размера вероятность события $V(e, \bar{\sigma}, s') = 1$ было пренебрежимо мала. Здесь $\sigma = C(e)$ и $s' = D(e, s)$, где $s = S(d, \sigma)$ (мы опускаем параметр n).

Теорема 36. *Если существует односторонняя функция, то существует протокол подписи одного бита.*

Доказательство. Пусть f_n односторонняя функция с трудной случайной величиной α_n . В качестве закрытого ключа возьмем пару $\langle x^0, x^1 \rangle$, где x^0, x^1 получены независимыми испытаниями случайной величины α_n . В качестве открытого ключа возьмем $\langle y^0, y^1 \rangle$, где $y^0 = f_n(x^0)$ и $y^1 = f_n(x^1)$.

Подпись под битом $\sigma = 0$ есть x^0 (первый компонент закрытого ключа), а подпись под $\sigma = 1$ есть x^1 (второй компонент закрытого ключа). Проверяющий применяет f_n к подписи s и принимает подпись, если $f_n(s) = y^\sigma$. Этот протокол удовлетворяет требованию (1). Проверим требование (2).

Допустим, что существуют последовательности схем C_n и D_n полиномиального от n размера, атакующие для бесконечно многих n построенную систему подписи с вероятностью успеха не меньше $\varepsilon = 1/\text{poly}(n)$. Зафиксируем любое из таких n . Противник, применяя схему C_n к открытому ключу y^0, y^1 вычисляет бит $\sigma = C(y^0, y^1)$. Затем, получив от подписывающего x^σ , он применяет схему D_n к открытому ключу и x^σ , получая поддельную подпись $\tilde{s} = D(y^0, y^1, x^\sigma)$. Атака будет успешной, если $y^\sigma = f(\tilde{s})$. Представим событие $y^\sigma = f(\tilde{s})$ в виде объединения двух событий

$$C(y^0, y^1) = 1 \wedge y^0 = f(D(y^0, y^1, x^1)), \quad (8)$$

$$C(y^0, y^1) = 0 \wedge y^1 = f(D(y^0, y^1, x^0)), \quad (9)$$

соответствующих $\sigma = 0$ и $\sigma = 1$. Вероятность хотя бы одного из этих событий не меньше $\varepsilon/2$. Пусть, скажем, это верно для первого события. Тогда вероятностная схема $D(y, f(x^1), x^1)$ обращает $f(\alpha)$ с вероятностью не менее $\varepsilon/2$. Действительно, вероятность того, что схема $D(y, f(x^1), x^1)$ обращает $f(\alpha)$ равна вероятности того, что $D(y^0, f(x^1), x^1)$ обращает y^0 , поскольку y^0 и $f(\alpha)$ одинаково распределены (при любом фиксированном x^1). А вероятность этого не меньше вероятности события (8). Осталось заметить, что вероятностная схема $D(y, f(x^1), x^1)$ имеет полиномиальный от n размер, поскольку функция f вычислима за полиномиальное время. \square

11.2 Подпись одного сообщения фиксированной длины

Протоколом подписи одного сообщения длины $p(n)$ (где p полином), называется последовательность доступных случайных величин $\langle d_n, e_n \rangle$ (закрытый и открытый ключи) и пара детерминированных алгоритмов S, V , называемыми алгоритмом подписи и алгоритм проверки, соответственно. Оба алгоритма получают на вход параметр безопасности n и работают время, ограниченное полиномом от n . Алгоритм P получает на вход,

кроме параметра безопасности, ключ d и последовательность битов x длины $p(n)$ и выдает подпись s . Алгоритм V получает на вход, кроме параметра безопасности, ключ e , слово x и подпись s и выдает 0 или 1 (отвергает или принимает подпись). Требования таковы:

(1) Для всех x длины $p(n)$ с вероятностью, приблизительно равной 1, выполнено $V(e, x, S(d, x)) = 1$ (параметр n мы опускаем). Вероятность берется по распределению на паре ключей (e, d) .

(2) Мы требуем защищенности протокола от следующей атаки. Противник, изучив открытый ключ e , просит Подписывающего подписать некоторое сообщение x . Подписывающий дает ему требуемую подпись $s = S(d, x)$. Противник изучает s и печатает некоторое сообщение x' , отличное от x и выдает поддельную подпись s' под ним. Итак, требование состоит в следующем. Для любой последовательности схем C_n, D_n, E_n полиномиального от n размера вероятность события

$$x' \neq x, \quad V(e, x', s') = 1$$

пренебрежимо мала (опускаем параметр n). Здесь $x = C(e)$, $x' = E(e, s)$ и $s' = D(e, s)$, где $s = S(d, x)$.

Теорема 37. *Если существует протокол подписи одного бита, то для любого полинома p существует протокол подписи одного сообщения длины $p(n)$.*

Доказательство. Пусть $\langle e_n, d_n \rangle$ и S, V данный нам протокол подписи одного бита. Рассмотрим “ $p(n)$ -ую степень этого протокола”, определяемую так: $\langle \tilde{e}_n, \tilde{d}_n \rangle = \langle e_n^1 \dots e_n^{p(n)}, d_n^1 \dots d_n^{p(n)} \rangle$ есть случайная величина, полученная $p(n)$ независимыми испытаниями случайной величины $\langle e_n, d_n \rangle$. Подпись \tilde{s} под сообщением $x_1 \dots x_{p(n)}$ состоит из конкатенаций подписей $s_1, \dots, s_{p(n)}$ с помощью S под битами $x_1, \dots, x_{p(n)}$, причем при подписывании бита x_i используется ключ d_n^i . Алгоритм проверки заключается в проверке всех подписей $s_1, \dots, s_{p(n)}$ с открытыми ключами $e_n^1, \dots, e_n^{p(n)}$. Если все они выдержали проверку, то подпись признается, иначе отвергается.

Условие (1) очевидно выполнено. Условие (2) проверяется так. Нам нужно атакующего $\tilde{C}_n, \tilde{D}_n, \tilde{E}_n$ новую систему подписи преобразовать в атакующего исходную систему. Пусть вероятность успешной атаки $\tilde{C}_n, \tilde{D}_n, \tilde{E}_n$ равна $\varepsilon = 1/\text{poly}(n)$ для бесконечно многих n . Очевидно существует $i \leq p(n)$, для которого с вероятностью не меньше $\varepsilon/p(n)$ атака будет

успешной и при этом i -ые биты x' и x различны. Успешность атаки означает, в частности, что $V(e^i, x'[i], s_i) = 1$ (параметр n опускаем). Будем атаковать исходную систему подписи одного бита следующим образом. Нам дан открытый ключ e , причем закрытый ключ, соответствующий ему, неизвестен. Выбираем случайным образом $p(n) - 1$ пару ключей. Даем полученную последовательность из $p(n) - 1$ открытых ключей на вход схеме \tilde{C} , вставив в нее на i -ое место данный нам открытый ключ e . Схема \tilde{C} выдаст некоторое сообщение x . Подписываем все его биты, кроме i -ого, с помощью известных закрытых ключей. А i -ый бит просим подписать Подписывающего один бит с ключом d (неизвестным нам). Полученную последовательность из n подписей даем на вход схеме D . Она выдаст n поддельных подписей. Оставляем в этой последовательности только i -ую подпись s'_i . Эта подпись с вероятностью не менее $\varepsilon/p(n)$ будет принята Проверяющим V , как подпись под $x'[i] = \overline{x[i]}$ с ключом e .

Описанная атака использует случайные биты (при выборе $p(n) - 1$ пар ключей). Зафиксировав их подходящим образом, мы можем получить детерминированные схемы с не меньшей вероятностью успеха. \square

Эта система подписи является “одноразовой” в следующем смысле. Имея подписи под сообщениями $00 \dots 0$ и $11 \dots 1$, можно изготовить подпись под любым сообщением. Как построить схему подписи, которую можно использовать любое полиномиальное число раз (и что это означает), мы обсудим позже.

11.3 Подпись одного сообщения произвольной длины (определение)

Протоколом подписи одного сообщения произвольной полиномиальной длины называется последовательность доступных случайных величин $\langle d_n, e_n \rangle$ (закрытый и открытый ключи) и пара алгоритмов S, V , называемыми алгоритмом подписи и алгоритм проверки, соответственно. Алгоритм S вероятностный, он получает на вход параметр безопасности n , ключ d и двоичную строку x и за время $\text{poly}(n + |x|)$ выдает за время $\text{poly}(n + |x|)$ подпись s . Алгоритм V детерминированный, он получает на вход параметр безопасности n , ключ e , слово x и подпись s и за время, ограниченное полиномом от $n + |x|$, выдает 0 или 1 (отвергает или принимает подпись). Требования таковы:

(1) Для любой последовательности слов x_n (длина слова x_n должна быть ограничена полиномом от n) с вероятностью, приблизительно равной 1, выполнено $V(e, x, S(d, x)) = 1$ (параметр n мы опускаем). Вероятность берется по распределению на паре ключей (e, d) и по исходам случайных бросаний алгоритма S .

(2) Мы требуем защищенности протокола от следующей атаки. Противник, изучив открытый ключ e , просит Подписывающего подписать некоторое сообщение $x = C(e)$ полиномиальной от n длины. Подписывающий дает ему требуемую подпись $s = S(d, x)$. Противник изучает ее и открытый ключ e и выдает сообщение x' и фальшивую подпись s' . Атака считается успешной, если $x' \neq x$ и подпись s' под x' принята алгоритмом V . Итак, требование состоит в следующем. Для любой последовательности схем C_n, D_n, E_n полиномиального от n размера вероятность события

$$x' \neq x, \quad V(e, x', s') = 1$$

пренебрежимо мала. Здесь $x = C(e)$, $x' = E(e, s)$, и $s' = D(e, s)$, где $s = S(d, x)$.

Схему электронной подписи одного сообщения произвольной полиномиальной длины можно построить, имея одноразовую систему подписи сообщений фиксированной длины с помощью так называемых семейств хэш-функций.

11.4 Семейства хэш-функций с трудно обнаружимыми коллизиями

Говоря неформально, хэш-функцией называется функция $h : \{0, 1\}^* \rightarrow \{0, 1\}^k$, для которой трудно найти такие $x_1 \neq x_2$, что $h(x_1) = h(x_2)$. Любая такая пара слов $\langle x_1, x_2 \rangle$ называется коллизией функции h . Поскольку мощность множества значений меньше мощности области определения, коллизии обязательно есть, но мы хотим, чтобы их было трудно обнаружить. Чтобы формализовать это понятие, будем говорить не об одной хэш-функции, а о семействе хэш-функций. Семейством хэш-функций называется полиномиально вычислимое отображение $h : \{0, 1\}^l \times \{0, 1\}^* \rightarrow \{0, 1\}^k$. При каждом фиксированном t из $\{0, 1\}^l$ получаем функцию $x \mapsto h(t, x)$, которую мы будем обозначать через h_t . Пусть семейство функций h зависит от натурального параметра n (параметра безопасности) и l, k являются полиномами от n , то есть задана последовательность отобра-

ражений $h^n : \{0, 1\}^{l(n)} \times \{0, 1\}^* \rightarrow \{0, 1\}^{k(n)}$. Пусть еще имеется доступная случайная величина α_n , принимающая значения в $\{0, 1\}^{l(n)}$. Пару, состоящую из h^n и α_n и называют семейством хэш-функций. Говорят, что семейство хэш-функций является семейством с трудно обнаруживаемыми коллизиями (СТОК), если для любой последовательности схем C_n размера $\text{poly}(n)$ вероятность события “ $C_n(\alpha_n)$ есть коллизия для $h_{\alpha_n}^n$ ” пренебрежимо мала.

Для преобразования системы подписи сообщений длины $k(n)$ в систему подписи сообщений любой длины достаточно хэш-функций, значения которых имеют $k(n)$ битов. Новый алгоритм подписи будет сначала применять хэш-функцию, а потом с помощью старого алгоритма подписывать полученное хэш-значение.

Теорема 38. *Если существует СТОК $h^n : \{0, 1\}^{l(n)} \times \{0, 1\}^* \rightarrow \{0, 1\}^{k(n)}$ и система одноразовой подписи сообщений из $k(n)$ битов, то существует и система одноразовой подписи сообщений произвольной длины.*

Доказательство. Пусть S, V данные нам в условии алгоритмы подписи и проверки, e_n, d_n — открытый и закрытый ключи для нее, а α_n — случайная величина из определения СТОК $h^n : \{0, 1\}^{l(n)} \times \{0, 1\}^* \rightarrow \{0, 1\}^{k(n)}$.

Нам нужно построить одноразовую систему подписи $(\langle \tilde{e}_n, \tilde{d}_n \rangle, \tilde{S}, \tilde{V})$ сообщения любой длины. В качестве открытого ключа возьмем пару $\langle e_n, \alpha_n \rangle$, состоящую из старого открытого ключа и идентификатора хэш-функции из семейства h_α . Аналогично определим закрытый ключ: $\tilde{d}_n = \langle d_n, \alpha_n \rangle$.

Алгоритм подписи \tilde{S} : сначала применяем к данному сообщению x хэш-функцию h_α , а затем подписываем старым алгоритмом S с ключом d полученное слово. То есть $\tilde{S}(\tilde{d}, x) = S(d, h_\alpha(x))$.

Алгоритм проверки \tilde{V} : сначала применяем к данному сообщению x хэш-функцию h_α , а затем проверяем старым алгоритмом проверки данную подпись, как подпись под полученным словом. То есть $\tilde{V}(\tilde{e}, x, s) = V(d, h_\alpha(x), s)$.

Очевидно, что требование (1) выполнено. Проверим условие (2). Рассуждая от противного, допустим, что имеются последовательности схем C_n, E_n, D_n полиномиального размера, для которых для бесконечно многих n вероятность успеха атаки не меньше $\varepsilon = 1/\text{poly}(n)$. То есть с этой вероятностью случается событие

$$T_n = (x' \neq x \wedge V(e, h_\alpha(x'), s') = 1),$$

где $x = C(e, \alpha)$, $x' = E(e, \alpha, s)$, и $s' = D(e, \alpha, s)$, где $s = S(d, h_\alpha(x))$. Вероятность здесь берется по случайному выбору e, d, α и случайным датчикам внутри E, S .

Будем атаковать семейство хэш-функций или схему подписи $(\langle e, d \rangle, S, V)$. От чего зависит, что именно мы атакуем? Одно из двух событий

$$A_n = (x' \neq x \wedge h_\alpha(x') = h_\alpha(x))$$

и

$$B_n = (h_\alpha(x') \neq h_\alpha(x) \wedge V(e, h_\alpha(x'), s') = 1)$$

имеет вероятность не менее $\varepsilon/2$. Действительно, вместе они покрывают событие T_n , которое имеет вероятность не менее ε . Либо для бесконечно многих n вероятность A_n не меньше $\varepsilon/2$, либо для бесконечно многих n вероятность B_n не меньше $\varepsilon/2$ (либо и то, и другое). В первом случае мы получим противоречие с тем, что семейство хэш-функций является СТОК, а во втором, получим противоречие с надежностью данной нам системы подписи.

Семейство хэш-функций атакуется следующим образом (в первом случае). Получив идентификатор α хэш-функции, мы запускаем алгоритм генерации пары e, d из данной нам системы подписи. Затем находим сообщение $x = C(e, \alpha)$ и подписываем его (поскольку мы сами сгенерировали оба ключа, мы можем подписать, что угодно), вычисляя $s = S(d, h_\alpha(x))$, а затем находим $x' = E(e, h_\alpha(x), s)$. Вероятность успеха этой атаки равна вероятности того, что $x' \neq x$, но $h_\alpha(x') = h_\alpha(x)$, то есть вероятности события A_n . По условию она не меньше $\varepsilon/2$ (для бесконечно многих n). Здесь важно, что распределение на парах (e, d, α) , которое мы используем, такое же, как и в определении события A_n .

Исходная система подписи $(\langle e, d \rangle, S, V)$ атакуется следующим образом (во втором случае). Получив открытый ключ e , мы запускаем алгоритм генерации идентификатора хэш-функции α , данный нам в условии. Затем находим сообщение $x = C(e, \alpha)$. Подписать его самостоятельно мы теперь не можем, поскольку у нас нет закрытого ключа. Поэтому мы просим обладателя закрытого ключа подписать $h_\alpha(x)$ (мы имеем право попросить один раз подписать сообщение длины $k(n)$). Получив $s = S(d, h_\alpha(x))$, мы вычисляем $x' = E(e, \alpha, s)$ и $s' = D(e, \alpha, s)$. В качестве результата мы выдаем сообщение $h_\alpha(x')$ и фальшивую подпись s' под ним. Вероятность успеха этой атаки равна вероятности того, что $h_\alpha(x') \neq h_\alpha(x)$, но $V(e, h_\alpha(x'), s') = 1$, то есть вероятности события B_n .

Здесь опять важно, что распределение на парах (e, d, α) , которое мы используем, такое же, как и в определении события B_n . \square

Эта система подписи обладает одним важным достоинством, не отраженным в определениях. Длина подписи зависит только от параметра безопасности (и не зависит от длины исходного сообщения).

11.5 Построение семейства хэш-функций с трудно обнаружимыми коллизиями

Для этого нам понадобится вспомогательный примитив — семейства перестановок с трудно обнаружимыми зацеплениями (claw-free). Зацеплением функций f^0, f^1 называется пара слов z, y , для которых $f^0(z) = f^1(y)$. Разрешается, чтобы z, y совпадали. Неформально, пара полиномиально вычислимых перестановок f_t^0, f_t^1 одного и того же множества D_t называется семейство с трудно обнаружимыми зацеплениями (СТОЗ), если по t трудно найти зацепление функций f_t^0, f_t^1 .

Пусть задана полиномиально вычислимая последовательность отображений $f_n : \{0, 1\} \times \{0, 1\}^{l(n)} \times \{0, 1\}^{m(n)} \rightarrow \{0, 1\}^{k(n)}$. Фиксируя n и последовательность t из $l(n)$ битов, мы получим функции $f_t^0(x) = f_n(0, t, x)$ и $f_t^1(x) = f_n(1, t, x)$ (параметр безопасности в обозначениях опускаем). Пусть еще имеется доступная последовательность случайных величин α_n , принимающая значения в $\{0, 1\}^{l(n)}$. И кроме того, для каждого возможного значения t случайной величины α_n имеется множество $D_t \subset \{0, 1\}^{m(n)}$, на котором обе функции f_t^0, f_t^1 являются перестановками. При этом требуется, чтобы существовал полиномиальный алгоритм, который по n и α с приблизительно единичной вероятностью находит некоторый элемент β_α из D_α . Вероятность здесь берется по данному нам распределению случайной величины α_n .

Будем такое семейство функций называть СТОЗ, если для любой последовательности схем C_n полиномиального размера вероятность того, что C_n на входе α выдает пару слов $y, z \in D_\alpha$, являющуюся зацеплением функций, f_α^0, f_α^1 , пренебрежимо мала. Вероятность здесь берется по данному нам распределению случайной величины α_n .

Семейство с трудно обнаружимыми зацеплениями можно построить, если предположить необратимость функции Рабина.

Теорема 39. *Если функция Рабина необратима, то существует СТОЗ.*

Доказательство. Случайная величина α_n равномерно распределена среди всех (m, x) из области определения функции Рабина ($m = pq$, где p, q — простые n -битовые числа вида $4k + 3$, а x — обратимый квадратичный вычет по модулю m). Множество $D_{m,x}$ состоит из всех обратимых квадратичных вычетов по модулю m . Функция $f_{m,x}^0(y)$ определена как $xy^2 \bmod m$, а $f_{m,x}^1(y) = y^2 \bmod m$ (вторая функция от x не зависит). Пары m, x из области определения функции Рабина можно генерировать с почти равномерным распределением, причем по m можно найти элемент из $D_{m,x}$ за полиномиальное время (скажем, $1 \in D_{m,x}$). Поэтому все исходные условия выполнены.

Проверим условие трудности обнаружения зацеплений. Допустим, мы можем не с пренебрежимо малой вероятностью по паре (m, x) из области определения функции Рабина находить такие обратимые квадратичные вычеты y, z , для которых $xy^2 \bmod m = z^2 \bmod m$. Поскольку вычет $(z/y) \bmod m$ является квадратичным и его квадрат равен x , с этой же вероятностью мы можем обращать функцию Рабина. \square

Задача 42. Докажите, что существует СТОЗ, если функция RSA необратима или экспонента в поле вычетов по простому модулю необратима.

Осталось построить СТОК, исходя из СТОЗ. Пусть дано семейство перестановок f_t^0, f_t^1 множества D_t с трудно обнаружимыми зацеплениями, причем распределение на t дается доступной случайной величиной α . Сопоставим каждому слову $x = x_1x_2 \dots x_m$ его префиксный код $\hat{x} = x_1x_1x_2x_2 \dots x_mx_m01$. Этот код обладает следующим свойством: если $x \neq z$, то ни одно из слов \hat{x}, \hat{z} не является началом другого. Рассмотрим следующее семейство хэш-функций

$$h_t(x) = f_t^{\hat{x}_1}(f_t^{\hat{x}_2}(\dots f_t^{\hat{x}_l}(\beta_t) \dots)),$$

где $\hat{x}_1\hat{x}_2 \dots \hat{x}_l$ обозначает префиксный код x . Слово t выбирается по распределению α , данному в определении СТОК.

Теорема 40. Построенное семейство функций является семейством с трудно обнаружимыми коллизиями.

Доказательство. Допустим, что схема S полиномиального размера с не пренебрежимо малой вероятностью находит коллизии у $h_t(x)$. То есть, она выдает пару слов $x \neq z$, для которых

$$f_t^{\hat{x}_1}(f_t^{\hat{x}_2}(\dots f_t^{\hat{x}_l}(\beta_t) \dots)) = f_t^{\hat{z}_1}(f_t^{\hat{z}_2}(\dots f_t^{\hat{z}_m}(\beta_t) \dots)).$$

Пусть i наименьшее число такое, что $\hat{x}_i \neq \hat{z}_i$ (такое существует по свойству префиксного кода). Поскольку f_t^0, f_t^1 — перестановки D_t и $\hat{x}_1 = \hat{z}_1, \dots, \hat{x}_{i-1} = \hat{z}_{i-1}$, мы можем сократить это равенство слева на композицию функций $f_t^{\hat{x}_1}, \dots, f_t^{\hat{x}_{i-1}}$:

$$f_t^{\hat{x}_i}(f_t^{\hat{x}_{i+1}}(\dots f_t^{\hat{x}_l}(\beta_t) \dots)) = f_t^{\hat{z}_i}(f_t^{\hat{z}_{i+1}}(\dots f_t^{\hat{z}_m}(\beta_t) \dots)).$$

Значит слова $f_t^{\hat{x}_{i+1}}(\dots f_t^{\hat{x}_l}(\beta_t) \dots)$ и $f_t^{\hat{z}_{i+1}}(\dots f_t^{\hat{z}_m}(\beta_t) \dots)$ являются зацеплением функций f_t^0, f_t^1 . \square

11.6 Универсальные семейства односторонних хэш-функций

Для протоколов электронной подписи, на самом деле, достаточно защищенности хэш-семейства от более слабой атаки. А именно, первое слово из коллизии противник должен указать, не зная хэш-функции, к которой подбирается коллизия.

Полиномиально вычислимое отображение $h^n : \{0, 1\}^{q(n)} \times \{0, 1\}^{p(n)} \rightarrow \{0, 1\}^n$ (здесь $p(n), q(n)$ — некоторые многочлены) называется универсальным семейством односторонних хэш-функций, если для любой последовательности схем C_n размера $\text{poly}(n)$ и любой последовательности слов $x_n \in \{0, 1\}^{p(n)}$ вероятность события «пара слов $\langle x_n, C_n(s) \rangle$ есть коллизия для h_s^n » пренебрежимо мала. Здесь s выбирается случайно среди слов длины $q(n)$.

Универсальное семейство односторонних хэш-функций существует при более слабых предположениях — достаточно существования любой односторонней функции [3].

11.7 Построение схемы подписи одного сообщения произвольной длины, исходя из универсального одностороннего семейства хэш-функций

Пусть имеется схема подписи $\langle \langle e_n, d_n \rangle, S, V \rangle$ одного сообщения длины $n + 1$. Пусть длина открытого ключа e_n в этой схеме равна полиному $p(n)$. Пусть, кроме того, имеется универсальное семейство односторонних хэш-функций $h^n : \{0, 1\}^{q(n)} \times \{0, 1\}^{p(n)} \rightarrow \{0, 1\}^n$, где $p(n)$ есть длина открытого ключа в схеме подписи, а $q(n)$ — произвольный многочлен.

Сначала построим систему подписи в которой условие $x' \neq x$ успеха атаки заменено на более сильное условие: ни одно из слов x' и x не является началом другого. Открытый ключ состоит из пары $\langle e, s \rangle$, а закрытый ключ — из пары $\langle d, s \rangle$ (опускаем параметр безопасности). Подпись под последовательностью битов $x = \sigma_1 \dots \sigma_l$ устроена так. Подписывающей выбирает случайно и независимо новые пары ключей $\langle e_2, d_2 \rangle, \dots, \langle e_{l+1}, d_{l+1} \rangle$. Затем подписывает (алгоритмом S) последовательность $h_s(e_2)\sigma_1$ с ключом d , последовательность $h_s(e_3)\sigma_2$ с ключом d_2 , и так далее. Последовательность $h_s(e_{i+1})\sigma_i$ подписывается с ключом d_i . Последней подписывается последовательность $h_s(e_{l+1})\sigma_l$ с ключом d_l (ключ e_{l+1} использоваться не будет). Обозначим через s_1, \dots, s_l полученную последовательность подписей. Общей подписью будет последовательность открытых ключей e_2, \dots, e_{l+1} и последовательность подписей s_1, \dots, s_l . Алгоритм проверки подписи состоит в применении алгоритма V с ключами e, e_2, \dots, e_l к сообщениям (соответственно) $h_s(e_2)\sigma_1, \dots, h_s(e_l)\sigma_{l-1}, h_s(e_{l+1})\sigma_l$ и подписям s_1, \dots, s_l .

Теорема 41. Построенные пара алгоритмов \tilde{S}, \tilde{V} и случайная величина $\langle \tilde{e}, \tilde{d} \rangle$ являются протоколом электронной подписи одного сообщения произвольной длины.

Доказательство. Пусть противник атакует построенную систему подписи, используя схемы $\tilde{C}, \tilde{D}, \tilde{E}$. То есть, сначала он вычисляет

$$x = \sigma_1 \dots \sigma_l = \tilde{C}(\alpha, e).$$

Затем получает подпись $\langle s_1 s_2 \dots s_l, e_2 e_3 \dots e_{l+1} \rangle$ под x , где

$$\begin{aligned} s_1 &= S(d, h_s(e_2)\sigma_1), \\ s_2 &= S(d_2, h_s(e_3)\sigma_2), \\ &\dots \\ s_l &= S(d_l, h_s(e_{l+1})\sigma_l). \end{aligned}$$

После этого он сам вычисляет

$$x' = \sigma'_1 \dots \sigma'_{l'} = \tilde{E}(s, e, s_1 s_2 \dots s_l)$$

и фальшивую подпись

$$s' = \langle s'_1 s'_2 \dots s'_{l'}, e'_2 e'_3 \dots e'_{l'+1} \rangle = \tilde{D}(s, e, s_1 s_2 \dots s_l).$$

Атака успешна, если x' не согласовано с x и

$$\begin{aligned} V(e, h_s(e'_2)\sigma'_1, s'_1) &= 1, \\ V(e_2, h_s(e'_3)\sigma'_2, s'_2) &= 1, \\ &\dots \\ V(e_{l'}, h_s(e_{l'+1})\sigma'_{l'}, s'_{l'}) &= 1. \end{aligned}$$

Допустим для бесконечно многих почти всех n вероятность этого события больше $\varepsilon = 1/\text{poly}(n)$.

Лемма 42. *Если атака успешна, то найдется такое $j \leq \min\{l, l'\}$, что $e_j \neq e'_j$, но $h_s(e_j) = h_s(e'_j)$, или найдется такое $j \leq \min\{l, l'\}$, что $e_j = e'_j$, но $\sigma_j h_s(e_{j+1}) \neq \sigma'_j h_s(e'_{j+1})$.*

В формулировке этой леммы мы предполагаем, что $e_1 = e'_1 = e$. Эта лемма утверждает, что у противника есть только две возможности подделывать подпись в новой системе: первая заключается в том, чтобы отыскать e'_j отличное от e_j с тем же хэш-значением, что и у e_j . Вторая возможность — подделать подпись в старой системе под сообщением $\sigma'_j h_s(e'_{j+1})$, отличным от сообщения $\sigma_j h_s(e_{j+1})$, подписанного владельцем секретного ключа d_j .

Доказательство. Допустим, что утверждение леммы неверно. В частности, оно неверно для $j = 1$. Поскольку $e_1 = e'_1 = e$, мы можем заключить, что $\sigma_1 h_s(e_2) = \sigma'_1 h_s(e'_2)$, то есть σ_1 и σ'_1 совпадают и $h_s(e_2) = h_s(e'_2)$. Пользуясь тем, что оно неверно для $j = 2$, выводим, что $e_2 = e'_2$. Рассуждая по индукции, мы установим, что первые $m = \min\{l, l'\}$ битов x и x' совпадают и $e_{m+1} = e'_{m+1}$. Это означает, что одно из слов x и x' является началом другого, то есть, атака не является успешной. \square

По лемме с вероятностью не меньше $\varepsilon/2$ то найдется такое $j \leq \min\{l, l'\}$, что $e_j \neq e'_j$, но $h_s(e_j) = h_s(e'_j)$, или найдется такое $j \leq \min\{l, l'\}$, что $e_j = e'_j$, но $\sigma_j h_s(e_{j+1}) \neq \sigma'_j h_s(e'_{j+1})$ и $V(e'_j, \sigma'_j h_s(e'_{j+1}), s'_j) = 1$. В первом случае, мы можем успешно атаковать семейство хэш-функций. Поскольку j ограничено некоторым полиномом $r(n)$ для некоторого фиксированного $j \leq r(n)$ с вероятностью не менее $\varepsilon/2r(n)$ будет выполнено $j \leq \min\{l, l'\}$, $e_j \neq e'_j$, но $h_s(e_j) = h_s(e'_j)$. Сгенерируем первый элемент коллизии, взяв случайный ключ e_j . Получив случайный номер хэш-функции s , сгенерируем случайную пару ключей e, d и дадим ее противнику, использующему схемы $\tilde{C}, \tilde{D}, \tilde{E}$. Он попросит нас подписать некоторое сообщение

x . Мы подписываем его, генерируя нужное количество пар ключей. Когда же он выдаст фальшивую подпись s' , возьмем в ней j -ый ключ e'_j . Этот ключ и будет с вероятностью не меньше $\varepsilon/2r(n)$ вторым элементом коллизии h_s . Описанная атака на семейство хэш-функций вычисляется вероятностными схемами полиномиального размера. Зафиксировав в этих схемах подходящим образом используемые случайные биты, получим детерминированные атакующие схемы полиномиального размера.

Пусть теперь с вероятностью не меньше $\varepsilon/2$ найдется такое $j \leq \min\{l, l'\}$, что $e_j = e'_j$, но $\sigma_j h_s(e_{j+1}) \neq \sigma'_j h_s(e_{j+1})$. Опять, это событие имеет место с вероятностью не менее $\varepsilon/2r(n)$ для некоторого фиксированного $j \leq r(n)$. Будем атаковать систему подписи одного сообщения длины $n + 1$. Пусть нам дан открытый ключ, обозначим его через e_j (он будет использован как j -ый ключ в подписи). Выберем случайно пару ключей e, d и хэш-функцию h_s . Затем дадим ключ $\tilde{e} = \langle e, s \rangle$ на вход схеме \tilde{C} . Подпишем выданное схемой сообщение $\sigma_1 \dots \sigma_l$, генерируя новые пары ключей $e_2, d_2, \dots, e_l, d_l$ (кроме j -ой пары). А подпись под $\sigma_j h_s(e_{j+1})$ получим у Подписывающего одно сообщение длины $n+1$ закрытым ключом d_j . Все подписи даем на вход схемам \tilde{D}, \tilde{E} . Схемы выдадут новое сообщение $\sigma'_1 \dots \sigma'_l$ и фальшивую подпись $\langle s'_1 s'_2 \dots s'_l, e'_2 e'_3 \dots e'_{l+1} \rangle$, про которые известно, что с вероятностью не менее $\varepsilon/2r(n)$ выполнено $e_j = e'_j$, $\sigma_j h_s(e_{j+1}) \neq \sigma'_j h_s(e'_{j+1})$ и $V(e'_j, \sigma'_j h_s(e'_{j+1}), s'_j) = 1$. То есть, $\sigma'_j h_s(e'_{j+1})$ является сообщением, отличным от того, которое мы просили подписать, и подпись s'_j под которым принимается с ключом $e_j = e'_j$.

Осталось построить схему в которой условие $x' \neq x$ успеха атаки не заменено на более сильное условие. Определим для каждого двоичного слова y его префиксный код \hat{y} следующим образом. Удвоим все биты y и припишем 01 в конце, например, $011 = 00111101$. По y легко найти \hat{y} , и если одно из слов вида \hat{y} является началом другого слова такого же вида, то эти слова совпадают. Модифицируем построенную систему подписи следующим образом. Будем вместо подписывания слова y подписывать его префиксный код \hat{y} (и при проверке будем проверять, подписано ли \hat{y}). Если два слова вида \hat{y} различны, то ни одно из них не является началом другого. Поэтому модифицированная система подписи удовлетворяет требованию (2) в его исходном виде. \square

11.8 Общий случай: подпись произвольного количества сообщений произвольной длины

Протоколом подписи (произвольного количества сообщений произвольной длины) называется последовательность доступных случайных величин $\langle d_n, e_n \rangle$ (закрытый и открытый ключи) и пара алгоритмов S, V , называемыми алгоритмом подписи и алгоритм проверки, соответственно. Алгоритм S вероятностный, он получает на вход параметр безопасности n , ключ d и двоичную строку x выдает за время $\text{poly}(n + |x|)$ подпись s . Алгоритм V детерминированный, он получает на вход параметр безопасности n , ключ e , слово x и подпись s и за время, ограниченное полиномом от $n + |x|$, выдает 0 или 1 (отвергает или принимает подпись). Требования таковы:

(1) Для любой последовательности слов x_n (длина слова x_n должна быть ограничена полиномом от n) с вероятностью, приблизительно равной 1, выполнено $V(e, x, S(d, x)) = 1$ (параметр n мы опускаем). Вероятность берется по распределению вероятностей на паре ключей (e, d) и по бросаниям монетки, выполняемым алгоритмом S .

(2) Мы требуем защищенности протокола от следующей атаки. Противник, изучив открытый ключ e , просит Подписывающего подписать некоторое сообщение $x_1 = C(e)$. Подписывающий дает ему требуемую подпись $s_1 = S(d, x_1)$. Противник изучает ее и просит Подписывающего подписать другое сообщение $x_2 = C(e, s_1)$. Подписывающий опять дает ему требуемую подпись $s_2 = S(d, x_2)$. Так продолжается k раз, где k ограничено полиномом от n . Изучив все k подписей s_1, \dots, s_k противник выдает сообщение $x_{k+1} = C(e, s_1 \dots s_k)$ и фальшивую подпись $s_{k+1} = D(e, s_1 \dots s_k)$. Атака считается успешной, если x_{k+1} не равно ни одному из сообщений x_1, \dots, x_k и подпись s_{k+1} под ним принята алгоритмом V . Итак, требование состоит в следующем. Для любой неравномерно полиномиальной стратегии C_n , любой неравномерно полиномиальной функции D_n и для любого полинома p для почти всех n вероятность события

$$x_{k+1} \notin \{x_1, \dots, x_k\}, \quad V(e, x_{k+1}, s_{k+1}) = 1$$

пренебрежимо мала. Здесь $k = p(n)$, а $x_1, \dots, x_k, x_{k+1}, s_{k+1}$ определяются

рекурсивно (мы опускаем параметр n):

$$\begin{aligned}
x_1 &= C(e), & s_1 &= S(d, x_1), \\
x_2 &= C(e, s_1), & s_2 &= S(d, x_2), \\
&\dots \\
x_k &= C(e, s_1, \dots, s_{k-1}), & s_k &= S(d, x_k), \\
x_{k+1} &= C(e, s_1, \dots, s_k), & s_{k+1} &= D(e, s_1, \dots, s_k).
\end{aligned}$$

Протокол подписи произвольного числа сообщений можно изготовить, исходя из протокола одноразовой подписи и семейства псевдослучайных функций. А именно, пусть $(\langle e_n, d_n \rangle, S, V)$ — одноразовая схема подписи сообщений любой длины. Пусть $l(n)$ — полином, ограничивающий количество случайных битов, нужных алгоритму генерации ключей e_n, d_n . Пусть $f_s^n : \{0, 1\}^* \rightarrow \{0, 1\}^{l(n)}$ — семейство ПСФ. В новой схеме $(\langle \tilde{e}_n, \tilde{d}_n \rangle, \tilde{S}, V)$ ключ \tilde{d} есть пара $\langle d, s \rangle$, где s задает псевдослучайную функцию, а d закрытый ключ из старой схемы. Ключ d выбирается независимо от s так, как положено в старой схеме, а s выбирается равномерно среди строк соответствующей длины. В открытый ключ включим только старый ключ: $\tilde{e} = e$.

Далее, определим $\tilde{S}(\tilde{d}, x)$ следующим образом. Определим *идентификатор* сообщения x , как его префиксный код. Идентификатор имеет длину $m = 2|x| + 2$, его биты мы будем обозначать через $\alpha_1 \dots \alpha_m$.

Рассмотрим $m-1$ вспомогательную пару ключей e_β, d_β для старой схемы, где β пробегает все непустые префиксы последовательности α . Пара e_β, d_β получается применением генератора ключей к последовательности битов $f_s(\beta)$ вместо используемой им случайной последовательности. Положим также пару (e_Λ, d_Λ) равной исходной паре ключей (e, d) (где Λ обозначает пустое слово). Новая подпись под сообщением x состоит из старой подписи с ключом d под парой ключей e_0, e_1 , старой подписи с ключом d_{α_1} под парой ключей $e_{\alpha_1 0}, e_{\alpha_1 1}$, старой подписи с ключом $d_{\alpha_1 \alpha_2}$ под парой ключей $e_{\alpha_1 \alpha_2 0}, e_{\alpha_1 \alpha_2 1}$ и так далее, всех подписанных ключей и, наконец, старой подписи с ключом $d_{\alpha_1 \dots \alpha_m}$ под самим сообщением x :

$$\begin{aligned}
\tilde{S}(\tilde{d}, x) &= (S(d, e_0 e_1), e_0, e_1, S(d_{\alpha_1}, e_{\alpha_1 0} e_{\alpha_1 1}), e_{\alpha_1 0}, e_{\alpha_1 1} \dots, \\
&S(d_{\alpha_1 \dots \alpha_{m-1}}, e_{\alpha_1 \dots \alpha_{m-1} 0} e_{\alpha_1 \dots \alpha_{m-1} 1}), e_{\alpha_1 \dots \alpha_{m-1} 0}, e_{\alpha_1 \dots \alpha_{m-1} 1}, S(d_{\alpha_1 \dots \alpha_m}, x)).
\end{aligned}$$

Проверяется эта подпись так: зная идентификатор, мы находим те старые ключи из новой подписи, которые нужно использовать для проверки

и с их помощью проверяем все старые подписи. Первую из старых подписей проверяем с помощью изначально данного нам старого ключа. То есть, \tilde{V} на тройке

$$\langle e, \alpha s_0 e_0^1 e_1^1 s_1 e_0^2 e_1^2 \dots s_m e_0^m e_1^m s_m, x \rangle$$

выдает 1, если

$$\begin{aligned} V(e, s_0, e_0^1 e_1^1) &= 1, \\ V(e_{\alpha_1}^1, s_1, e_0^2 e_1^2) &= 1, \\ &\dots, \\ V(e_{\alpha_{m-1}}^{m-1}, s_{m-1}, e_0^m e_1^m) &= 1, \\ V(e_{\alpha_m}^m, s_m, x) &= 1. \end{aligned}$$

Отметим сразу важное свойство построенной схемы: при любой атаке для всех β ключ d_β в ней используется только один раз. Действительно, если β является идентификатором некоторого сообщения x , то ключ d_β используется только для подписи слова x , а если β является собственным началом идентификатора некоторого сообщения, то ключ d_β используется только для подписи слова $e_{\beta_0} e_{\beta_1}$. В силу выбора идентификаторов, никакое слово не может быть одновременно некоторым идентификатором и собственным началом некоторого идентификатора.

Теорема 43. *Построенная схема $(\langle \tilde{e}_n, \tilde{d}_n \rangle, \tilde{S}, \tilde{V})$ является схемой подписи произвольного числа сообщений.*

Доказательство. Условие (1) очевидно выполнено. Проверим условие (2). Пусть имеется схема C_n , которая для бесконечно многих n атакует новую схему с вероятностью успеха $\varepsilon = 1/\text{poly}(n)$. Заменим в новой схеме ПСФ f_s на случайную функцию $f : \{0, 1\}^* \rightarrow \{0, 1\}^{l(n)}$. Подписывающий алгоритм \tilde{S} вместе с атакующей схемой C и проверяющим алгоритмом \tilde{V} можно рассматривать как вероятностную схему для проверки случайности функции f_s : имея функцию f в качестве оракула, эта схема генерирует ключи e, d и моделирует атаку схемы C на новую систему подписи с ключами $(d, s), e$ (заметим, что s при моделировании атаки знать не нужно), затем схема выдает 1, если атака прошла успешно. В силу надежности семейства ПСФ, в результате подмены вероятность успеха атаки изменится незначительно, поэтому для бесконечно многих

n она не меньше $\varepsilon/2$. Таким образом, мы можем считать далее, что алгоритм \tilde{S} выбирает все ключи (e_β, d_β) независимо друг от друга, используя алгоритм генерации ключей из старой схемы.

Будем сравнивать поддельную подпись

$$s_{k+1} = (\hat{s}_0, \hat{e}_0^1, \hat{e}_1^1, \hat{s}_1, \hat{e}_0^2, \hat{e}_1^2, \dots, \hat{s}_m, \hat{e}_0^m, \hat{e}_1^m, \hat{s}_m)$$

под сообщением x_{k+1} с настоящей подписью

$$\begin{aligned} \tilde{S}(e, d, x) = & (S(d, e_0 e_1), e_0, e_1, S(d_{\alpha_1}, e_{\alpha_1 0} e_{\alpha_1 1}), e_{\alpha_1 0}, e_{\alpha_1 1} \dots, \\ & S(d_{\alpha_1 \dots \alpha_{m-1}}, e_{\alpha_1 \dots \alpha_{m-1} 0} e_{\alpha_1 \dots \alpha_{m-1} 1}), e_{\alpha_1 \dots \alpha_{m-1} 0}, e_{\alpha_1 \dots \alpha_{m-1} 1}, S(d_{\alpha_1 \dots \alpha_m}, x)). \end{aligned}$$

Напомним, что x_{k+1} и s_{k+1} изготовлены схемой после изучения настоящих подписей под сообщениями x_1, \dots, x_k . Возможно одно из двух: либо существует такое $1 \leq i \leq m$, что $\hat{e}_{\alpha_i}^i \neq e_{\alpha_1 \dots \alpha_i}$ (то есть атакующий подменил хотя бы один из ключей, использованных в настоящей подписи), либо нет. В первом случае рассмотрим наименьшее такое i . Во втором случае положим $i = m + 1$. Допустим сначала, что нам заранее известно i и даже известны первые $i - 1$ битов $\alpha_1 \dots \alpha_{i-1}$ идентификатора слова x_{k+1} . То есть атакующий заранее решает, какой именно из открытых ключей он собирается атаковать и сообщает нам идентификатор $\alpha_1 \dots \alpha_{i-1}$ этого ключа, изучив открытый ключ e .

Если атака успешна, то фальшивая подпись s_{k+1} успешно прошла проверку и при этом x_{k+1} не равно ни одному из подписанных сообщений x_1, \dots, x_k . Поэтому \hat{s}_{i-1} является правильной подписью с правильным ключом под сообщением, которое владелец закрытого ключа d не подписывал. То есть, $V(e_{\alpha_1 \dots \alpha_{i-1}}, \hat{s}_{i-1}, \hat{e}_0^i \hat{e}_1^i) = 1$, если $1 \leq i \leq m$, и $V(e_{\alpha_1 \dots \alpha_m}, \hat{s}_m, x_{k+1}) = 1$, если $i = m + 1$.

Рассмотрим следующий алгоритм атаки на старую схему подписи. Получив открытый ключ, мы даем его схеме C и она нам сообщает $\alpha_1 \dots \alpha_{i-1}$, а также x_1 . Сначала допустим, что $\alpha_1 \dots \alpha_{i-1}$ не является началом идентификатора слова x_1 . Тогда мы генерируем ключи e_β, d_β для всех начал идентификатора слова x_1 , запоминаем их для будущего использования и с помощью них подписываем сообщение x . Если же $\alpha_1 \dots \alpha_{i-1}$ является началом идентификатора слова x_1 , то делаем то же самое, за исключением генерации пары $e_{\alpha_1 \dots \alpha_i}, d_{\alpha_1 \dots \alpha_i}$. Вместо ключа $e_{\alpha_1 \dots \alpha_{i-1}}$ мы берем ключ e , а соответствующую подпись берем у владельца ключа d (мы имеем право спросить один раз подпись под любым словом). Затем передаем изготовленную подпись схеме C , получаем от нее x_2 и

повторяем процесс. Если вновь $\alpha_1 \dots \alpha_{i-1}$ окажется началом идентификатора слова x_2 , то это начало собственное (как для идентификатора x_2 , так и для идентификатора x_1). Поэтому просить вторично подпись у владельца ключа d нам не придется, поскольку мы уже знаем правильную подпись под сообщением $e_{\alpha_1 \dots \alpha_{i-1} 0} e_{\alpha_1 \dots \alpha_{i-1} 1}$. И так далее. При этом мы запоминаем все уже сгенерированные ключи, чтобы вторично не генерировать ключи e_β, d_β , если они уже сгенерированы ранее. Поэтому наши действия совпадают с действиями алгоритма \hat{S} , а значит вероятность того, что $V(e, \hat{s}_{i-1}, \hat{e}_0^i \hat{e}_1^i) = 1$, если $1 \leq i \leq m$, и $V(e, \hat{s}_m, x_{k+1}) = 1$, если $i = m + 1$, будет не меньше $\varepsilon/2$. В первом случае мы выдаем пару $\hat{s}_{i-1}, \hat{e}_0^i \hat{e}_1^i$, а во втором пару \hat{s}_m, x_{k+1} .

Осталось понять, как атаковать старую схему подписи, если схема C не сообщает нам заранее $\alpha_1 \dots \alpha_{i-1}$. Тогда мы пытаемся угадать i , а также первое j , для которого $\alpha_1 \dots \alpha_{i-1}$ является началом идентификатора x_j . Поскольку i и j заведомо не превосходят некоторых известных нам полиномов от n , вероятность правильного угадывания не меньше $1/\text{poly}(n)$. Атака в этом случае выглядит так: атакуем так же, как и раньше, при этом в качестве $\alpha_1 \dots \alpha_{i-1}$ используем первые $i - 1$ битов идентификатора x_j . Заметим, что в первый раз нам понадобится узнать $\alpha_1 \dots \alpha_{i-1}$ только после того, как нам сообщено x_j . Вероятность успеха этой атаки будет в полином от n раз меньше $\varepsilon/2$, чего нам достаточно. \square

Алгоритм подписи в построенной системе детерминирован. Однако она обладает важным недостатком — длина подписи растет с ростом длины сообщения. Пожертвовав детерминированностью алгоритма подписи, можно изготовить систему, в которой длина подписи зависит только от параметра безопасности. А именно, модифицируем алгоритмы подписи и проверки следующим образом. Алгоритм подписи в качестве идентификатора сообщения берет не его префиксный код, а случайную строку длины n , которую он включает в состав подписи. А алгоритм проверки использует вместо префиксного кода сообщения идентификатор, входящий в состав подписи.

Задача 43. Докажите, что так определенная система подписи надежна.

Задача 44. Модифицируйте систему подписи из теоремы 43 так, чтобы длина подписи зависела только от параметра безопасности и при этом алгоритм подписи остался детерминированным.

Список литературы

- [1] Eric Bach: How to Generate Factored Random Numbers. SIAM J. Comput. 17(2): 179-193 (1988)
- [2] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, Michael Luby: A Pseudorandom Generator from any One-way Function. SIAM J. Comput. (SIAMCOMP) 28(4):1364-1396 (1999)
- [3] Rompel.