Лекция 2

Семейство односторонних функций. Трудный бит.

(Конспект: В. Николаенко, Д.М. Ицыксон)

2.1 Семейство односторонних функций

Определение 2.1. *Семейством односторонних функций* называется детерминированный алгоритм G, который получает на вход параметр надежности n, вектор случайных битов r_g и за полиномиальное от n время выдает две булевы схемы e и s $(G : (1^n, r_g) \mapsto (e, s))$.

Первая схема задает функцию $e : \{0,1\}^n \to \{0,1\}^{\epsilon(n)}$, где $\epsilon(n)$ - полином. e — это одна из односторонних функций.

$s : \{0,1\}^{\sigma(n)} \to \{0,1\}^n$ - генератор трудных входов для e. $\sigma(n)$ - полином.

И выполняется условие: $\forall$ полинома $p(n)$ $\forall$ вероятностного полиномиального по времени алгоритма A $\forall$ достаточно больших n, выполняется:

$$\Pr[A(e(x)), 1^n) \in e^{-1}(e(x))] < \frac{1}{p(n)}$$

Вероятность берётся по следующим параметрам: $x = s(y)$, где y равномерно распределен на $\{0,1\}^{\sigma(n)}$, $(e, s) = G(1^n, r_g)$, а r_g равномерно распределен на $\{0,1\}^n$ и по случайным битам алгоритма A.

Утверждение 2.1. $\exists$ *односторонняя функция* $\Leftrightarrow$ $\exists$ *семейство односторонних функций*

Доказательство. Пусть f - односторонняя функция. Определим $G(1^n, r) = (e, s)$, где e — это схема для f на входах длины n, а $s(x) = x$. Нетрудно проверить, что построенный G — это семейство односторонних функций.

Пусть G - семейство односторонних функций, пусть g работает время, ограниченное полиномом $q(n)$. Можно считать, что длина второго параметра ровно $q(n)$ Определим функцию f только на входах длины $\log n + q(n) + \sigma(n)$ следующим образом: если

$G(1^{|t|}, r_g) = (e, s)$, то $f(t, r_g, r_s) = e(s(r_s)))$. Нетрудно проверить, что f — это односторонняя функция. □

2.2 Универсальная односторонняя функция

В этом параграфе мы предъявим функцию, которая будет слабо односторонней, если односторонние функции вообще существуют.

Лемма 2.1. *Если ∃ односторонняя функция, то ∃ односторонняя функция, которая вычислима за время* $200n^2$.

Доказательство. Пусть $f : \{0,1\}^n \to \{0,1\}^{p(n)}$ - односторонняя, вычислимая за время $q(n)$, тогда $\widetilde{f}(x,t) = f(x)$, $|t| = q(n), n = |x|$ - односторонняя, вычислимая за $200n^2$ □

Пусть $M_1, M_2, ...$ - нумерация машин Тьюринга, которые работают $200n^2$ шагов (можно считать, что в них встроен будильник, который прекращает работу машины Тьюринга через $200n^2$ шагов).

Будем считать, что результат работы машины — это $200n^2$ символов ленты (даже если машина была прервана будильником), пробелы можно отождествлять с нулями. Тогда любая машина задает функцию: $\{0,1\}^n \to \{0,1\}^{200n^2}$

Определение 2.2. *Универсальной функцией Левина* называется функция $g : (M, x) \mapsto (M, M(x))$, где M — это описание машины Тьюринга, а $M(x)$ — это результат работы машины M на входе x за не более, чем $200|x|^2$ шагов. Пару (M, x) можно записать, например, так: в записи машины M продублировать все биты, потом написать 01 и дальше x обычным способом.

Теорема 2.1. *Универсальная функция Левина является слабо односторонняя, если односторонние функции существуют.*

Доказательство. Поскольку односторонние функции существуют, то $\exists i_0$, что M_{i_0} вычисляет одностороннюю функцию. M_{i_0} — это конкретная машина, поэтому $|M_{i_0}|$ является константой.

Пусть универсальная функция Левина g не является односторонней, тогда для любого полинома $p(n)$ ∃ полиномиальный вероятностный алгоритм A, что для бесконечного числа длин n выполняется:

$$\frac{1}{2^{|M_{i_0}|}p(n)} > \Pr_{(M,x)\leftarrow U_n}[A(M, M(x), 1^n) \notin g^{-1}(M, M(x))]$$
$$\geq \frac{1}{2^{|M_{i_0}|}} \Pr_{x\leftarrow U_{n-|M_{i_0}|}}[A(M_{i_0}, M_{i_0}(x), 1^n) \notin g^{-1}(M_{i_0}, M_{i_0}(x))]$$

Если посмотреть на начало и конец цепочки неравенств, то получится, что для бесконечного числа n выполняется:

$$\Pr_{x \leftarrow U_{n-|M_{i_0}|}}[A(M_{i_0}, M_{i_0}(x), 1^n) \notin g^{-1}(M_{i_0}, M_{i_0}(x))] < \frac{1}{p(n)},$$

что противоречит тому, что M_{i_0} вычисляет слабо одностороннюю функцию. □

2.3 Трудный бит (Hard-Core Predicate)

Односторонняя функция f обладает таким свойством, что по $f(x)$ трудно найти x. Но вовсе необязательно, что по $f(x)$ трудно найти первый бит x, например, односторонняя функция может не менять первый бит.

Определение 2.3. Пусть $h : \{0,1\}^* \to \{0,1\}^*$ - полиномиально вычислимая функция. $b : \{0,1\}^* \to \{0,1\}$ - называется **трудным битом** (или трудным предикатом) для h, если $\forall$ многочлена $p(n)$ $\forall$ полиномиального вероятностного алгоритма A

$$\Pr_{x \leftarrow U_n}[A(h(x), 1^n) = b(x)] < \frac{1}{2} + \frac{1}{p(n)}.$$

Оказывается, что любую одностороннюю функцию можно немного модифицировать так, чтобы у нее появился трудный бит.

Теорема 2.2 (Голдрейх-Левин). *Пусть f сохраняющая длину односторонняя функция, пусть $g : g(x,r) = (f(x), r), |x| = |r|$. Тогда $b(x,r) = <x,r> = \bigoplus_{i=1}^{n} x_i r_i$ является трудным битом для функции g.*

Доказательство. Пусть $b(x,r)$ не является трудным битом, тогда $\exists$ алгоритм G $\exists$ многочлен $p(n)$: для всех достаточно больших n:

$$Pr_{(x,r) \leftarrow U_{2n}}[G(f(x), r) = b(x,r)] \geq \frac{1}{2} + \frac{1}{p(n)}.$$

Наша цель показать, что основываясь на алгоритме G, можно обратить функцию f.

Обозначим $s(x) = Pr_{r \leftarrow U_n}[G(f(x), r) = b(x,r)]$. Пусть $S_n = \{x \in \{0,1\}^n | s(x) \geq \frac{1}{2} + \frac{1}{2p(n)}\}$ — множество тех входов, на которых алгоритм G вычисляет $b(x,r)$ достаточно хорошо.

Покажем, что $|S_n| \geq 2^n \frac{1}{2p(n)}$. Действительно, пусть $|S_n| \leq 2^n \frac{1}{2p(n)}$, тогда

$$\begin{aligned}
\Pr_{(x,r) \leftarrow U_n}[G(f(x), r) = b(x,r)] &= \Pr[G(f(x), r) = b(x,r) \mid x \in S_n] Pr[x \in S_n] \\
&\quad + \Pr[G(f(x), r) = b(x,r) \mid x \notin S_n] Pr[x \notin S_n] \\
&< \Pr[x \in S_n] + Pr_{(x,r) \leftarrow U_{2n}}[G(f(x), r) = b(x,r) \mid x \notin S_n] \\
&< \frac{1}{2p(n)} + \left(\frac{1}{2} + \frac{1}{2p(n)}\right) = \frac{1}{2} + \frac{1}{p(n)},
\end{aligned}$$

противоречие с выбором G.

Предположим, что множество S_n определялось бы немного иначе: $\widetilde{S}_n = \{x \in \{0,1\}^n | s(x) \geq \frac{3}{4} + \frac{1}{2p(n)}\}$. Тогда для $x \in \widetilde{S}_n$ с вероятностью как минимум $\frac{1}{2} + \frac{1}{p(n)}$ для случайного $r \in \{0,1\}^n$ выполняется $G(f(x), r) = b(x, r)$ и одновременно $G(f(x), r \oplus e_i) = b(x, r \oplus e_i)$, где e_i — это строка из n нулей и единиц с единственной 1 на позиции с номером i. Таким образом мы можем с вероятностью $\geq \frac{1}{2} + \frac{1}{p(n)}$ вычислить i-ый бит строки x, который равен $<x, e_i> = <x, r> + <x, r \oplus e_i> = b(x, r) \oplus b(x, r \oplus e_i)$. Повторив полиномиальное число раз и взяв наиболее частый ответ мы могли бы вычислить i-й бит строки x с вероятностью $1 - 2^{-\Omega(n)}$. Однако, S_n у нас определяется не так удачно, и у нас нет возможности делать два запроса к алгоритму G, поскольку вероятность того, что ответы на оба запроса будут правильные меньше, чем $\frac{1}{2}$. Наша ближайшая цель сэкономить на одном запросе.

Чтобы избавиться от одного запроса к G, сделаем следующее:

Сгенерируем независимо ℓ случайных равномерно распределенных строчек длины n: $s_1, s_2, \ldots, s_l$. Для каждого $J \subset \{1, 2, \ldots, l\}, J \neq \emptyset$ определим свою строку $r_J = \bigoplus_{i \in J} s_i$. Число ℓ положим равным $\log_2(2np^2(n)+1)$, видно, что $\ell = O(\log n)$. Число строчек r_J полиномиальное от n. Кроме того сгенерируем независимо ℓ случайных битов $\sigma_1, \sigma_2, \ldots, \sigma_\ell$. С вероятностью $\frac{1}{2^\ell} = \frac{1}{\text{poly}(n)}$ выполняется $b(x, S_i) = \sigma_i$. В этом случае можно вычислить $b(x, r_J) = \bigoplus_{i \in J} \sigma_i$.

Упражнение 2.1. Покажите, что все r_J одинаково распределены и попарно независимы.

Покажем, что $\forall x \in S_n \forall 1 \leq i \leq n$ выполняется неравенство

$$\Pr[|J \subset \{1, 2, \ldots, l\}, J \neq \emptyset \mid b(x, r_J) \oplus G(f(x), r_J \oplus e_i) = x_i| > \frac{2^\ell - 1}{2}] > 1 - \frac{1}{2n}.$$

Другими словами с большой вероятностью для большинства множеств J алгоритм G работает правильно на входе $(f(x), r_J \oplus e_i)$.

Действительно, рассмотрим величину

$$X_J = \begin{cases} 1 & \text{когда } G(f(x), r_J \oplus e_i) = b(x, r_J \oplus e_i) \\ 0 & \text{иначе} \end{cases}$$

X_J попарно независимы и одинаково распределены, так как r_J попарно независимы и одинаково распределены. По свойству дисперсии получаем $\mathrm{D}[\sum X_J] = \sum \mathrm{D}[X_J] = (2^\ell - 1)\,\mathrm{D}[X_{J_0}]$, тут D - обозначает дисперсию. $\mathrm{D}[X_J] = \mathrm{E}[X_J^2] - \mathrm{E}[X_J]^2$, поскольку $\mathrm{E}[X_J^2] = \mathrm{E}[X_J] = \frac{1}{2} + \epsilon$ (тут $\epsilon > \frac{1}{2p(n)}$), то можно оценить дисперсию $\mathrm{D}[X_J] = \frac{1}{2} + \epsilon - (\frac{1}{2} + \epsilon)^2 < 1/4$.

Вспомним неравенство Чебышева:

$$Pr\left[|X - \mathrm{E}[X]| \geq k \cdot \sqrt{\mathrm{D}[X]}\right] \leq \frac{1}{k^2}$$

Получим:

$$\Pr\left[\sum X_J \le \frac{1}{2}(2^\ell - 1)\right] \le \Pr\left[|\sum X_J - (2^\ell - 1)\cdot(\frac{1}{2} + \frac{1}{2p(n)})| \ge \frac{(2^\ell - 1)}{2p(n)}\right] =$$
$$\Pr[|\sum X_J - (2^\ell - 1)\cdot(\frac{1}{2} + \frac{1}{2p(n)})| \ge \frac{\sqrt{\mathrm{D}[X_{J_0}](2^\ell - 1)}\sqrt{2^\ell - 1}}{2p(n)\sqrt{\mathrm{D}[X_{J_0}]}}] \le$$
$$\frac{\mathrm{D}[X_{J_0}]\cdot 4\cdot p^2(n)}{2^\ell - 1} \le \frac{p^2(n)}{2np^2(n)} = \frac{1}{2n}$$

Опишем теперь алгоритм A, который будет обращать f. Сначала A будет независимо выбирать случайные строки $s_1, s_2, \ldots, s_l$ и биты $\sigma_1, \ldots, \sigma_\ell$. Затем для каждого для каждого $1 \le i \le n$ алгоритм A переберет все непустые подмножества $J \subseteq \{1, \ldots, l\}$, для каждого из них посчитает $(\bigoplus_{i\in J}\sigma_i) \oplus (G(f(x), \bigoplus_{i\in J} s_i \oplus e_i)$, пусть α_i — это самый частый встречающийся ответ. Алгоритм A выдает на выход строку $\alpha_1\alpha_2\ldots\alpha_n$

Проверим, что с вероятностью $\frac{1}{poly(n)}$ алгоритм A обращает f:

1. С вероятностью $\frac{1}{2^\ell} = \frac{1}{poly(n)}$ для всех $1 \le j \le \ell$ выполняется $b(x, s_j) = \sigma_j$.
2. С вероятностью $1 - \frac{1}{2n}$ i-й бит x равен α_i.
3. С вероятностью $\ge \frac{1}{2}$ можно восстановить все биты $x = \alpha_1\alpha_2\ldots\alpha_n$. Заметим, что эту вероятность можно повысить повторениями, так как можно проверить, верно ли, что $f(\alpha_1\alpha_2\ldots\alpha_n) = f(x)$.
4. С вероятностью $\ge \frac{1}{2p(n)}, x \in S_n$

Итого, получим, что алгоритм A обратит $f(x)$ с вероятностью $\ge \frac{1}{poly(n)}$. Получили противоречие с тем, что f односторонняя функция.

Замечание 2.1. Вовсе необязательно угадывать биты $\sigma_1, \sigma_2, \ldots, \sigma_\ell$. Достаточно их всех перебрать (их количество полиномиально от n) и проверить ответ (вычислить $f(\alpha_1\alpha_2\ldots\alpha_n)$).

□