Лекция 7

IP = PSPACE. Лемма Вэлианта-Вазирани

(Конспект: С. Николенко)

7.1 IP = PSPACE

Интерактивные протоколы. **IP** (Interactive Protocol) — это класс языков, полиномиально распознаваемых при помощи так называемого интерактивного протокола. Интерактивный протокол — это алгоритм, описывающий процесс передачи данных между двумя вычислительными устройствами: P (Prover) и V (Verifier). P обладает неограниченными вычислительными возможностями, V — полиномиальная по времени вероятностная машина Тьюринга (P не имеет доступа к лентам V; в частности, к используемым V случайным числам). P пытается убедить V принять вход алгоритма, а V хочет принять его тогда и только тогда, когда этот вход принадлежит языку, для которого и составлен протокол. То есть P передает V некоторую информацию, а V ее проверяет в меру своих полиномиальных способностей. Язык L принадлежит **IP**, если существует такой полиномиальный по времени (и размеру передаваемых данных) протокол, что

- если $x \in L$, то P всегда может убедить V в том, что вход надо принять;
- если $x \notin L$, то P может убедить V с вероятностью не более $\frac{1}{2}$.

Пример 7.1. Язык, состоящий из пар неизоморфных графов, принадлежит **IP**: V берет выбирает случайным образом граф из пары, применяет к его вершинам случайную перестановку, отправляет полученный граф P и просит его отгадать, какой же граф он выбрал.

Ясно, что **IP** содержится в **PSPACE**: переберем все возможные ответы Prover'а; если для какого-то из них при моделировании Verifier'а (с перебором возможных значений случайных битов) все ветви будут принимающими, то $x \in L$. Значит, для доказательства равенства достаточно показать, что некоторый **PSPACE**-полный язык разпознаваем **IP**-протоколом. Мы будем использовать язык булевых формул с кванторами (`QBF`), то есть формул вида $Q_1x_1 \dots Q_nx_nB(x_1 \dots x_n)$, где $B(x_1 \dots x_n)$ — булева формула без кванторов, и $Q_1 \dots Q_n \in \{\forall, \exists\}$.

Арифметизация. Бескванторную булеву формулу $B(x_1 \dots x_n)$ можно «арифметизовать»: поставить ей в соответствие многочлен $b(x_1 \dots x_n)$, значение которого на 0 и 1 совпадает с соответствующим значением формулы B (где 0 соответствует `false`, 1 соответствует `true`). Именно, $\alpha \wedge \beta$ заменяется на $\alpha \cdot \beta$, $\neg\alpha$ — на $1 - \alpha$, и $\alpha \vee \beta$ — как $\neg(\neg\alpha \wedge \neg\beta)$ (т.е. на $1-(1-\alpha)(1-\beta)$ — эту формулу обозначим $\alpha * \beta$). Очевидно, что значение многочлена b, записанного согласно этим правилам в виде формулы (скобки не раскрываем!), вычислить просто — но для этого должны быть заданы значения (свободных) переменных.

Арифметизуем теперь всю формулу с кванторами. Пусть теперь $P(x, \dots)$ — некоторый многочлен (по смыслу — каким-то образом полученный нами из булевой формулы). Определим операторы «раскрытия» кванторов

$$(A_xP)(\dots) = P(0, \dots) \cdot P(1, \dots),$$

$$(E_xP)(\dots) = P(0, \dots) * P(1, \dots),$$

и оператор понижения степени (линеаризации)

$$(L_xP)(x, \dots) = P \mod (x^2 - x)$$

(то есть все x^n при $n > 1$ заменяются на x).

Многочлен L_xP имеет те же переменные, что и P; в A_xP и E_xP переменная x отсутствует. Отметим, что P и L_xP совпадают при значениях переменных из $\{0, 1\}$. Ясно, что исходная булева формула с кванторами

$$Q_1x_1 \dots Q_nx_nB(x_1 \dots x_n)$$

имеет то же самое значение, что и

$$q^{(1)}_{x_1} \quad L_{x_1} \quad q^{(2)}_{x_2} \quad L_{x_1}L_{x_2} \quad q^{(x_3)} \quad \dots \quad q^{(n)}_{x_n} \quad L_{x_1} \dots L_{x_n} \quad b(x_1 \dots x_n), \tag{7.1}$$

где $q^{(i)} = A$, если $Q_i = \forall$, $q^{(i)} = E$, если $Q_i = \exists$. Можно было бы обойтись и без операторов L_x; они применяются, чтобы степень промежуточных многочленов (при вычислении (7.1) справа налево) была не слишком велика.

Протокол. Наша задача — вычислить значение (7.1). Сделать это трудно (при раскрытии скобок будет получаться экспоненциально много мономов). Поэтому мы (Verifier) воспользуемся Prover'ом. При описании протокола заодно будем доказывать и его корректность, *выделяя это курсивом.*

Протокол будет заключаться в том, что мы будет последовательно спрашивать P о полиномах, являющихся не до конца вычисленным (7.1), снимая кванторы слева направо. Пусть $q \in \{E, A, L\}$, а R — полином, записанный как в (7.1), т.е. при помощи операторов E, A, L, а не в явном виде. Нашей промежуточной задачей будет убедиться в том, что

$$q_{x_i} R(x_1, \ldots, x_{i-1}, x_i) = c, \tag{7.2}$$

для некоторых заданных числовых значений свободных переменных $x_1 = r_1, \ldots, x_{i-1} = r_{i-1}$ (и $x_i = r_i$, если $q = L$). (Ясно, что исходная задача формулируется именно в таких терминах: $i = 1$, R — вся формула (7.1), кроме первого оператора q.) Мы будем сводить нашу задачу к задаче для более короткого R, т.е. наш протокол будет рекурсивным.

Благодаря операторам L, степень по каждой переменной промежуточных многочленов при вычислении (7.1) справа налево не превосходит d, где d — общее количество вхождений переменных в исходную формулу[1]. Многочленами $q_{x_i} R$ в нашем алгоритме как раз будут эти промежуточные многочлены, поэтому мы будем требовать, чтобы степень $R(r_1, \ldots, r_{i-1})$ как многочлена от одной переменной не превосходила d. Случайные числа будут выбираться из поля $\mathbb{F} = \mathbb{Z}/p\mathbb{Z}$ остатков по модулю простого числа p, которое следует взять чуть большим, чем d^4 (p может выбрать либо P, либо V, потому что проверка простоты для чисел такого размера тривиальна).

Строгое описание протокола (вернее, рекурсивной процедуры проверки того, что если произвести все действия в «многочлене» $q_{x_i} \ldots q'_{x_n} R(r_1, \ldots, r_{i-1}, x_i, \ldots, x_n)$ с внешним оператором q [сам многочлен записан как в условии, только вместо первых $i - 1$ операторов к нему прилагаются значения для соответствующих переменных], то получится константа c) таково. Имеется три случая:

$q = A$. P должен послать V коэффициенты многочлена от одной переменной $s(x_i) = R(r_1, \ldots, r_{i-1}, x_i)$. Если $\deg s > d$ или $s(0)s(1) \neq c$, V отвергает вход алгоритма (*и он прав — если действительно (7.2) верно и P прислал правильные коэффициенты s, равенство $s(0)s(1) = c$ было бы выполнено по определению оператора A*).

[1] На самом деле, такой большой степень может быть только для самого внутреннего многочлена b, а далее не превосходит 2.

В противном случае *остается убедиться, что коэффициенты многочлена s — правильные (если это так, то задача выполнена).* V выбирает случайное число $r_i \in F$. Теперь, рекурсивно используя тот же протокол, P должен убедить V, что $R(r_1 \dots r_{i-1}, r_i) = s(r_i)$. *Заметим, что вероятность того, что это равенство выполнено, но коэффициенты многочлена s — неправильные, не превосходит d/d^4 (мы случайно попали в корень многочлена степени не более d над полем размера $\geq d^4$). Этим исчерпывается возможность ошибиться на этом шаге — дальнейшее зависит только от рекурсивного вызова.*

$q = E$. Полностью аналогично предыдущему пункту (только проверять надо, конечно, что $s(0) * s(1) = c$).

$q = L$. P должен послать V коэффициенты многочлена от одной переменной $s(x) = (R(r_1 \dots r_{i-1}, x)$. Если $\deg s > d$ или $s(0) + (s(1) - s(0))r_i \neq c$, то V отвергает (*отметим, что* $s(0)+(s(1)-s(0))r_i$ — *это значение* $s(x) \mod (x^2 - x)$ *при* $x = r_i$ — *так что он снова правильно делает, что отвергает*).

Так мы убеждаемся, что линеаризация была произведена правильно. Снова *остается убедиться, что коэффициенты многочлена s — правильные.* Тогда V выбирает случайный элемент $r'_i \in \mathbb{F}$. Теперь, рекурсивно используя тот же протокол, P должен убедить V, что $R(r_1, \dots, r_{i-1}, r'_i) = s(r'_i)$. *Вероятность ошибиться на этом шаге – снова d/d^4.*

Таким образом, мы совершим $O(d^2)$ рекурсивных вызовов, причем в каждом из них вероятность ошибиться не превосходит $1/d^3$. Таким образом, вероятность ошибки нашего протокола $\leq O(d^2) \cdot 1/d^3$.

7.2 Лемма Вэлианта-Вазирани.

Лемма Вэлианта-Вазирани является одним из первых нетривиальных результатов об отношениях между сложностными классами.

Лемма 7.1 (L. Valiant, V. Vazirani)**.** *Имеется вероятностное (с односторонней ошибкой) полиномиальное по времени сведе́ние задачи выполнимости к ее индивидуальным задачам, имеющим не более одного выполняющего набора.*

Иными словами, по заданной формуле F в КНФ можно построить формулы $F_1, \dots, F_m$ в КНФ, такие что

- если F выполнима, то с вероятностью, большей $1/2$, по крайней мере одна из формул $F_1, \ldots, F_m$ имеет в точности один выполняющий набор;
- если F невыполнима, то все формулы $F_1, \ldots, F_m$ невыполнимы.

Пусть F — формула, а набор A присваивает значения всем ее переменным. Отождествим A с n-битным числом $a = a_0 a_1 \ldots a_{n-1}$, таким что $a_j = 1$, если соответствующая переменная в наборе A имеет значение `true`, и $a_j = 0$ в противном случае. Выберем целые числа p_i и r_i следующим образом. Сначала выберем равновероятно случайным образом целое $i \in [0..n]$. Затем выберем равновероятно случайным образом $p_i \in [1..b_i]$ и $r_i \in [0..b_i]$, где $b_i = 4 \cdot 2^i n^2$. Заменим F на формулу

$$F \wedge (a \mod p_i = r_i).$$

Выражение «$(a \mod p_i = r_i)$» обозначает здесь булеву формулу в КНФ от переменных $a_0, \ldots, a_{n-1}$ (возможно, также использующую дополнительные переменные, причем значения дополнительных переменных *однозначно* определяются по значениям остальных переменных), которая представляет соответствующее арифметическое сравнение. Например, эта формула может быть получена посредством кодирования обычного умножения «в столбик»[2]. Очевидно, что это сведе́ние полиномиально по времени и переводит невыполнимую формулу в невыполнимую формулу. Остается доказать, что если F выполнима, то с большой вероятностью новая формула одновыполнима.

Пусть $a^{(1)}, \ldots, a^{(D)}$ — все выполняющие наборы формулы F. Заметим, что $i = \lceil \log_2 D \rceil$ с вероятностью $1/(n+1)$. Предположим, что это событие имеет место. Заметим, что для данных j, h $(j \neq h)$ имеется не более n простых делителей разности $a^{(j)} - a^{(h)}$. С другой стороны, для достаточно больших n имеется, как минимум, $0.92129 \cdot b_i / \ln b_i > b_i / \log_2 b_i \geq 2^{i+1} n$ простых чисел, не превосходящих b_i. Таким образом, имеется, по крайней мере, $2^{i+1}n - 2^i n = 2^i n$ чисел p, не превосходящих b_i, таких, что остаток выполняющего набора $a^{(j)}$ по модулю p отличается от остатков всех других выполняющих наборов по модулю p. Следовательно, по крайней мере $2^i n$ пар $0 \leq p_i, r_i \leq b_i$ «отличают» набор $a^{(j)}$ от всех остальных. Заметим, что для различных выполняющих наборов множества «отличающих» пар дизъюнктны. Следовательно, имеется не менее $2^i n \cdot D \geq 2^{2i-1} n$ искомых пар. Таким образом, для достаточно больших n вероятность выбрать такую пару составляет, по крайней мере, $\frac{2^{2i-1}n}{16 \cdot 2^{2i} n^4} = \frac{1}{32n^3}$.

[2]Элементарные операции над битами кодируются так: чтобы определить новый бит $z = x \wedge y$, допишем дизъюнкции $\neg x \vee \neg y \vee z$, $\neg z \vee x$, $\neg z \vee y$. Остальные операции можно закодировать при помощи этой и отрицания (а можно и напрямую).

Домножая на вероятность выбрать «правильное» i, получаем, что вероятность ошибки в нашем сведéнии не превосходит $1-\frac{1}{32n^4+32}$. Выбирая тройки (i, p_i, r_i) случайным образом $O(n^4)$ раз, получаем константную вероятность ошибки.