Краткий конспект лекций курса
Алгоритмические методы в теории сложности.
А.Е.Ромащенко. Мехмат МГУ, весна 2013.

Предлагаемые заметки состоят из конспектов лекций спецкурса, прочитанного в весеннем семестре 2013 года на мехмате МГУ. Разные лекции записаны с разной степенью подробности: для некоторых лекций записаны полные доказательства теорем; в других случаях лишь напоминается краткий план прочитанной лекции. В конспекте не обсуждаются мотивировки и содержательный смысл излагаемых теорем. Данный конспект будет полезен студентам, посещавшим лекции, но едва ли может быть использован как самостоятельное учебное пособие.

Последние исправления: 12 июля 2013 г.

1 Лекция 1, 22 февраля.

1.1 Лемма об изоляции [НО]

Пусть $\mathcal{F}$ — некоторое семейство подмножеств из универсума $U = \{1, \dots, n\}$ (каждое множество $S \in \mathcal{F}$ есть *непустое* подмножество $S \subset U$). Будем придавать элементам универсума *веса* $w : U \to \{1, \dots, T\}$ (вес каждого элемента есть целое число от 1 до некоторого порога T). Для каждого распределения весов w будем называть весом множества S сумму весов его элементов:

$$w(S) = \sum_{x \in S} w(x).$$

Лемма 1 (Первая лемма об изоляции.)**.** *Зафиксируем универсум U, семейство непустых подмножеств универсума $\mathcal{F}$ и число T. Будем выбирать случайное распределение весов $w : U \to \{1, \dots, T\}$ (веса всех элементов выбираются независимо и равномерно из множества $1, \dots, T$). Тогда с вероятностью не менее $1 - n/T$ в $\mathcal{F}$ найдется ровно одно множество минимального веса.*

Доказательство: Для каждого элемента x рассмотрим разность

$$\alpha(x) = \min_{S : S \not\ni x} w(S) - \min_{S : S \ni x} w(S)$$

(все множества S берутся из семейства $\mathcal{F}$). Простое наблюдение:

- если $\alpha(x) < 0$, то минимальный вес имеет некоторое множество $S \in \mathcal{F}$, которое не содержит x;
- если $\alpha(x) > 0$, то минимальный вес имеет некоторое множество $S \in \mathcal{F}$, которое содержит x;

- если $\alpha(x) = 0$, то минимальный вес достигается не менее чем на двух разных множествах из $\mathcal{F}$; одно из этих множеств содержит x, а другое нет.

Если минимум веса достигается сразу на двух разных множествах $S_1, S_2 \in \mathcal{F}$, то найдется такой элемент x, для которого $\alpha(x) = 0$ (нужно взять x, которое принадлежит разности $S_1 \setminus S_2$ или в $S_2 \setminus S_1$). И наоборот, если $\alpha(x) = 0$ хотя бы для одного x, то в семействе $\mathcal{F}$ есть не менее двух разных множеств с минимальным весом. Таким образом, минимум веса достигается ровно на одном множестве S, если и только если $\alpha(x)$ не равно нулю ни для какого x.

Чтобы оценить вероятность интересующего нас события, удобно немного преобразовать выражение $\alpha(x)$. Рассмотрим для каждого $x \in U$ величину

$$\beta(x) = \min_{S : S \not\ni x} w(S) - \min_{S : S \ni x} (w(S) - w(x)).$$

Сделанное выше наблюдение можно переформулировать так:

- если $\beta(x) < w(x)$, то минимальный вес имеет некоторое множество $S \in \mathcal{F}$, которое не содержит x;
- если $\beta(x) > w(x)$, то минимальный вес имеет некоторое множество $S \in \mathcal{F}$, которое содержит x;
- если $\beta(x) = w(x)$, то минимальный вес достигается не менее чем на двух разных множествах из $\mathcal{F}$; одно из этих множеств содержит x, а другое нет.

Таким образом, минимум веса достигается сразу на нескольких множествах из $\mathcal{F}$, если и только если$\beta(x) = w(x)$ для некоторого элемента x. Вероятность этого события оценить нетрудно. Воспользуемся тем, что $\beta(x)$ зависит от весов всех элементов универсума *кроме* x. Это значит, что $\beta(x)$ и $w(x)$ независимы. Поскольку значение случайные величины $w(x)$ распределено равномерно на $\{1, \ldots, T\}$, для каждого числа ω

$$\mathrm{Prob}[w(x) = \beta(x) \mid \beta(x) = \omega] = \begin{cases} 1/T, & \text{если } 1 \le \omega \le T, \\ 0, & \text{иначе.} \end{cases}$$

Следовательно, для каждого $x \in U$

$$\mathrm{Prob}[\beta(x) = w(x)] \le 1/T.$$

Суммируя вероятности таких событий по всем x, получаем

$$\mathrm{Prob}[\exists x \ \beta(x) = w(x)] \le n/T,$$

и лемма доказана.

Лемма 2 (Вторая лемма об изоляции)**.** *Для любых натуральных чисел n и k ($n \geq k$) существует такое семейство $\mathcal{H}_{n,k}$, состоящее из $2^{O(n)}$ функций*

$$h : \{0,1\}^n \to \{0,1\}^k,$$

для которого выполнено следующее условие. Для любого множества $S \subset \{0,1\}^n$ такого, что $2^{k-1} \leq |S| \leq 2^{k-1}$, выполняется

$$\mathrm{Prob}_{h \in \mathcal{H}}[\text{существует ровно одно } x \in S \text{ для которого } h(x) = 0 \ldots 0] \geq 1/8$$

(*т.е., для многих функций h из семейства $\mathcal{H}$ есть ровно один элемент из x, который под действием h переходит в последовательность из k нулей*).

Более того, требуемое семейство $\mathcal{H}_{n,k}$ можно построить явно, и значение каждой функции $h_i \in \mathcal{H}_{n,k}$ можно вычислять за полиномиальное время (*по индексу функции i и заданному значению аргумента*).

Набросок доказательства: Довольно легко построить семейство функций, обладающее всеми требуемыми свойствами, но имеющего при этом размер несколько больше требуемого — семейство из $2^{O(n^2)}$ функций. Для этого достаточно рассмотреть семейство всех аффинных отображений

$$h : \mathbb{F}_2^n \to \mathbb{F}_2^k,$$

которые представляются в виде

$$h(\bar{x}) = A \cdot \bar{x} + \bar{b},$$

где $\bar{x}$ обозначает n-битный вектор-столбец (вектор из n-мерного пространства над полем из двух элементов), A обозначает матрицу $k \times n$ над тем же полем (матрица задаёт линейное отображение из n-мерного пространства в k-мерное), а $\bar{b}$ обозначает вектор сдвига (вектор-столбец из k-мерного пространства над полем из двух элементов). Нетрудно убедиться, что для данного семейства функций выполнены два свойства:

- *равномерность:* для каждого $x \in \mathbb{F}_2^n$ и каждого $y \in \mathbb{F}_2^k$

$$\mathrm{Prob}_{h \in \mathcal{H}}[h(x) = y] = 1/2^k,$$

- 2-*независимость:* для любых $x_1, x_2 \in \mathbb{F}_2^n$ ($x_1 \neq x_2$) и любых $y_1, y_2 \in \mathbb{F}_2^k$

$$\mathrm{Prob}_{h \in \mathcal{H}}[h(x_1) = y_1 \text{ и } h(x_2) = y_2] = (1/2^k)^2.$$

Пусть теперь множество S имеет требуемый размер (не менее 2^{k-2} и не более 2^{k-1} элементов). Выберем в описанном семействе случайную аффинную функцию $h \in \mathcal{H}$ и обозначим $N = N(h)$ число таких $x \in S$, что $h(x) = 0 \ldots 0$.

Замечание: Из свойства *равномерности* распределения значений h следует, что математическое ожидание $E_h N = |S| \cdot \frac{1}{2^k}$.

Далее, по формуле включений-исключений

$$\mathrm{Prob}[N \geq 1] \geq \sum_{x \in S} \mathrm{Prob}[h(x) = 0 \dots 0] - \sum_{x \neq x' \in S} \mathrm{Prob}[h(x) = h(x') = 0 \dots 0] =$$

$$= |S| \cdot \frac{1}{2^k} - C^2_{|S|} \cdot \left(\frac{1}{2^k}\right)^2 \geq \left(\frac{|S|}{2^k}\right) - \frac{1}{2} \cdot \left(\frac{|S|}{2^k}\right)^2.$$

С другой стороны,

$$\mathrm{Prob}[N \geq 2] \leq \sum_{x \neq x' \in S} \mathrm{Prob}[h(x) = h(x') = 0 \dots 0] = C^2_{|S|} \cdot \left(\frac{1}{2^k}\right)^2 \leq \frac{1}{2} \cdot \left(\frac{|S|}{2^k}\right)^2.$$

В итоге получаем

$$\mathrm{Prob}_{h \in \mathcal{H}}[N = 1] = \mathrm{Prob}[N \geq 1] - \mathrm{Prob}[N \geq 2] \geq \left(\frac{|S|}{2^k}\right) - \left(\frac{|S|}{2^k}\right)^2 \geq \frac{1}{8}.$$

Мы почти доказали лемму – построенное семейство функций $\mathcal{H}$ удовлетворяет всем требованиям, но пока оно слишком большого размера. Остаётся придумать, как уменьшить размер множества отображений с $2^{O(n^2)}$ до $2^{O(n)}$, не потеряв полезных для нас свойств. Стандартное решение состоит в том, что в качестве A нужно брать не все матрицы размера $k \times n$, а только такие, у которых на каждой диагонали, параллельной главной, стоит какое-то одно значение. Более формально, мы рассматриваем такие матрицы, что если $i - j = i' - j'$, то матричные элементы a_{ij} и $a_{i'j'}$ равны друг другу. Чтобы задать такую матрицу A, достаточно указать её элементы в первой строке и в первом столбце (по одному элементу на каждой из диагоналей). Так что каждое отображение из нового семейства функций (все аффинные отображения вида $h(\bar{x}) = A \cdot \bar{x} + \bar{b}$, где A является матрицей указанного вида) однозначно задаётся $O(n)$ битами, и число всех таких отображений равно $2^{O(n)}$.

Несложно проверить, что свойства *равномерности* и 2-*независимости* выполнены для уменьшенного множества матриц. Следовательно, и требуемое свойство «изоляции» для каждого множества S также выполнено.

1.2 Поиск совершенного паросочетания

На лекции мы использовали первую лемму об изоляции, чтобы свести задачу о поиске совершенного паросочетания в двудольном графе к задаче о вычислении определителя матрицы. В итоге мы получили полиномиальный вероятностный алгоритм нахождения совершенного паросочетания, который хорошо поддается распараллеливанию, см. [MVV].

2 Лекция 2, 1 марта.

На этой лекции мы обсуждали определение алгоритма и разные модели вычисления, понятие переборной задачи, классы P, BPP, NP, $\oplus$P, UP, #P,

см. [AB]

3 Лекция 3, 15 марта.

Обсуждение формализации «переборных» задач и напоминание определения сложностных классов P, BPP, NP, $\oplus$P, UP, #P.

Теорема 1. *Существует вероятностный полиномиальный алгоритм A, преобразующий булеву формулу φ в булеву формулу $\psi = A(\varphi)$ так, что*

- *если φ выполнима, то*

$$\mathrm{Prob}[\psi \text{ имеет ровно один выполняющий набор}] > 1/N,$$

- *если φ невыполнима, то*

$$\mathrm{Prob}[\psi \text{ имеет хотя бы один выполняющий набор}] = 0,$$

где $N = \mathrm{poly}(|\varphi|)$.

Доказательство теоремы состоит в применении одной из двух лемм об изоляции. Сначала изложим доказательство, основанное на второй леммы.

Пусть булева формула φ содержит n переменных $x_1, \ldots, x_n$. Обозначим множество выполняющих наборов этой формулы S (множество S является подмножеством $\{0,1\}^n$). Пусть множество S непусто. Тогда найдется такое целое k, что

$$2^{k-2} < |S| \leq 2^{k-1}.$$

Согласно Лемме 2 можно построить такое семейство функций $\mathcal{H}_{n,k}$, что для не менее, чем $1/8$ всех функций $h \in \mathcal{H}_{n,k}$ существует ровно один набор $x = (x_1, \ldots, x_n) \in S$, для которого $h(x) = 0 \ldots 0$.

Теперь нетрудно догадаться, как должен действовать вероятностный алгоритм A. Сначала он угадывает число k (если S непусто, то правильное значение k угадывается с вероятностью $1/n$). Далее алгоритм строит семейство $\mathcal{H}_{n,k}$ и выбирает из него случайную функцию h. Если значение k было угадано верно, то с вероятностью не менее $1/8$ для h выполняется свойство изоляции: существует ровно один элемент $x \in S$, для которого $h(x) = 0 \ldots 0$.

Таким образом, мы получаем свойство

$$(*) \qquad \varphi(x_1 m \ldots, c_n) \;\&\; h(x_1, \ldots, x_n) = 0 \ldots 0,$$

которому (при условии удачного выбора k и h) удовлетворяет ровно один набор булевых значений. Заметим также, что если исходная формула φ была невыполнимой, то при любом выборе k и h свойство (*) не выполнено ни на одном наборе x.

Осталась одна небольшая техническая трудность: свойство (*) пока не булева формула, а лишь некоторый n-местный предикат. Но (*) нетрудно

переписать в виде булевой формулы. Напомним, что функция h есть аффинное отображение из $\mathbb{F}_2^n$ в $\mathbb{F}_2^k$. Так что свойство $h(x_1,\ldots,x_n) = 0\ldots0$ можно записать в виде конъюнкции k условий, каждое из которых есть утверждение о чётности суммы нескольких булевых значений x_i (результат умножения $(x_1,\ldots,x_n)$ на одну из строк матрицы линейного преобразования h, с нулевым или с единичным сдвигом). Переписав правую часть (*) в виде булевой формулы, мы получаем требуемую формулу ψ. Первое доказательство теоремы закончено.

На лекции мы обсуждали и второе доказательство данной теоремы, использующее первую лемму об изоляции. Напомним, что в этом варианте доказательства итоговая формула ψ содержала кроме переменных $x_1,\ldots,x_n$ также несколько дополнительных переменных z_j.

Заметим, что приведенное выше рассуждение доказывает не только Теорему 1, но и следующее более общее утверждение.

Теорема 1′. *Существует вероятностный полиномиальный алгоритм A, который по заданному n строит (случайную) булеву формулу $\theta(x_1,\ldots,x_n)$ со следующим свойством. Для каждого предиката $F:\{0,1\}^n\to\{0,1\}$*

- *если существует набор булевых значений $(x_1,\ldots,x_n)$ такой, что $F(x_1,\ldots,x_n)=1$, то*

$$\mathrm{Prob}[\exists!\ x\ (F(x_1,\ldots,x_n)=1\ \&\ \theta(x_1,\ldots,x_n))] > 1/\mathrm{poly}(n),$$

- *если не существует набора булевых значений $(x_1,\ldots,x_n)$ таких, что $F(x_1,\ldots,x_n)=1$, то*

$$\mathrm{Prob}[\exists x\ (F(x_1,\ldots,x_n)=1\ \&\ \theta(x_1,\ldots,x_n))] = 0.$$

Доказательство: Прежде всего заметим, что в такой формулировке вторая часть утверждения («*если не существует набора булевых значений $(x_1,\ldots,x_n)$ таких, что $F(x_1,\ldots,x_n)=1$...*») тривиальна; мы сохранили его лишь для того, чтобы подчеркнуть аналогию с Теоремой 1.

Доказательство Теоремы 1′ почти дословно повторяет доказательство Теоремы 1. Разница лишь в том, что вместо булевой формулы $\varphi(x_1,\ldots,x_n)$ мы используем абстрактный предикат $F(x_1,\ldots,x_n)=1$ (не уточняя, как именно этот предикат определен).

Теорема 2. *Существует вероятностный полиномиальный алгоритм A', преобразущий граф G и натуральное число k в такие граф G' и число k', что*

- *если в G есть клика размера k, то*

$$\mathrm{Prob}[G'\ \textit{содержит ровно одну клику размера}\ k'] > 1/N,$$

- *если в G нет клики размера k, то*

$$\mathrm{Prob}[G'\ \textit{содержит хотя бы одну клику размера}\ k'] = 0,$$

где $N = \mathrm{poly}(|\varphi|)$.

Доказательство: применяем *первую* лемму об изоляции. См. доказательство в [MVV].

4 Лекция 4, 22 марта.

Введем следующее обозначение: если φ булева формула, то $\#\varphi$ обозначает число выполняющих наборов для φ, а $\oplus\varphi$ обозначает чётность числа выполняющих наборов для этой формулы (1, если число выполняющих наборов нечётно, и 0, если число выполняющих наборов чётно).

Далее нам потребуется ввести на булевых формулах операции $\cdot$, $+$, $\mathbin{⩒}$, $+1$ (для любой пары формул φ и ψ мы построим формулы $\varphi \cdot \psi$, $\varphi + \psi$, $\varphi \mathbin{⩒} \psi$; для каждой формулы φ мы построим соответствующую формулу $\varphi + 1$). Нам нужно, чтобы выполнялись следующие свойства:

$$\begin{aligned} \#(\varphi \cdot \psi) &= (\#\varphi) \cdot (\#\psi), \\ \#(\varphi + \psi) &= (\#\varphi) \cdot (\#\psi), \\ \#(\varphi + 1) &= \#(\varphi) + 1. \\ \oplus(\varphi \mathbin{⩒} \psi) &= (\oplus\varphi) \vee (\oplus\psi). \end{aligned}$$

Нетрудно придумать явную конструкцию преобразования формул с требуемыми свойствами. Нам будет важно, что данные операции над формулами можно производить алгоритмически. Формально нам потребуется следующее утверждение:

Утверждение 1. *Существует детерминированный полиномиальный алгоритм, который по заданным формулам φ и ψ строит $\varphi \cdot \psi$, $\varphi + \psi$, $\varphi + 1$, $\varphi \mathbin{⩒} \psi$.*

Доказательство: Будем предполагать, что в в φ и ψ не встречаются одни и те же переменные (этого всегда можно добиться, переименовав переменные в одной из формул).

(1) Для получения $\varphi \cdot \psi$ достаточно взять конъюнкцию двух формул.

(2) Для получения $\varphi(x_1, \ldots, x_n) + \psi(y_1, \ldots, y_m)$ можно взять формулу

$$(\varphi(x_1, \ldots, x_n) \ \&\ y_1 \& \ldots \ \&\ y_m \ \&\ z) \mathbin{⩒} (\psi(y_1, \ldots, y_m) \ \&\ x_1 \ \&\ \ldots \ \&\ x_n \ \&\ \neg z)$$

(3) В качестве $\varphi(x_1, \ldots, x_n) + 1$ можно взять формулу

$$(\varphi(x_1, \ldots, x_n) \ \&\ y_1 \ \&\ \ldots \ \&\ y_m \ \&\ z) \mathbin{⩒} (x_1 \ \&\ \ldots \ \&\ x_n \ \&\ \neg z).$$

(4) $\varphi(x_1, \ldots, x_n) \mathbin{⩒} \psi(y_1, \ldots, y_m)$ можно определить как

$$(\varphi + 1) \cdot (\psi + 1) + 1.$$

Нетрудно проверить, что данные определения соответствуют условиям утверждения.

Замечание: Утверждение 1 говорит, что множество языков в классе $\oplus$SAT замкнуто относительно объединений, пересечений и дополнений.

Теорема 3. *Существует вероятностный алгоритм B, который по заданной булевой формуле φ и числу m строит такую формулу $\psi = B(\varphi, m)$, что*

- *если φ выполнима, то* $\mathrm{Prob}[\psi \in \oplus\mathrm{SAT}] > 1 - 2^{-m}$,
- *если φ невыполнима, то* $\mathrm{Prob}[\psi \in \oplus\mathrm{SAT}] = 0$.

(Вероятность берется по случайным битам алгоритма.) Алгоритм B работает за время $\mathrm{poly}(|\varphi|, m)$.

Доказательство теоремы: По Теореме 1 существует вероятностный алгоритм A такой, что для некоторого $N = \mathrm{poly}(|\varphi|)$

- если φ выполнима, то $\mathrm{Prob}[\#(A(\varphi)) = 1] > 1/N$,
- если φ невыполнима, то $\mathrm{Prob}[\#(A(\varphi)) > 0] = 0$.

Эти два свойства можно переформулировать так:

- если φ выполнима, то $\mathrm{Prob}[\oplus(A(\varphi)) = 1] > 1/N$,
- если φ невыполнима, то $\mathrm{Prob}[\oplus(A(\varphi)) = 1] = 0$.

Применим к формуле φ алгоритм A из Теоремы 1 несколько раз, используя каждый раз свежие значения датчика случайных битов. Если мы применим алгоритм s раз, то получим в итоге s булевых формул

$$\psi_1, \ldots, \psi_s.$$

При этом мы можем утверждать, что

- если φ выполнима, то $\mathrm{Prob}[\oplus(\psi_i) = 1] > 1/N$ для каждого $i = 1, \ldots, s$,
- если φ невыполнима, то $\mathrm{Prob}[\oplus(\psi_i)) = 1] = 0$ для каждого $i = 1, \ldots, s$.

В первом случае (если формула φ выполнима), то вероятность того, что *ни одна* из полученных ψ_i не имеет нечетного числа выполняющих наборов, не превосходит $(1 - 1/N)^s$. Следовательно, если число вызовов алгоритма A достаточно велико ($s = const \cdot mN$), то мы можем утверждать, что

- если φ выполнима, то $\mathrm{Prob}[\exists i \ \oplus(\psi_i) = 1] > 1 - 2^{-m}$,
- если φ невыполнима, то $\mathrm{Prob}[\exists i \ \oplus(\psi_i)) = 1] = 0$.

Остается соединить все формулы ψ_i вместе. Для этого мы воспользуемся операцией $\barwedge$ и положим

$$\hat{\psi} := (\ldots(\psi_1 \barwedge \psi_2) \barwedge \psi_3) \barwedge \ldots \psi_s) \ldots).$$

По доказанному выше,

- если φ выполнима, то $\mathrm{Prob}[\oplus(\hat{\psi}) = 1] > 1 - 2^{-m}$,

- если φ невыполнима, то $\mathrm{Prob}[\oplus(\hat{\psi}) = 1] = 0$.

Таким образом, алгоритм нужный нам B должен действовать следующим образом: $s = O(mN)$ раз применить к формуле φ вероятностный алгоритм A из Теоремы 1, а затем объединить все полученные формулы с помощью операции $\circledcirc$. Теорема доказана.

Подобно тому, как Теорема 1 имела более абстрактный аналог Теорему $1'$, для Теоремы 3 имеется следующее обобщение:

Теорема 3′. *Существует вероятностный алгоритм B, который по заданным числам n и m строит такое формальное выражение выражение ψ, что для любого предиката $F : \{0,1\}^n \to \{0,1\}$*

- *если существует такой набор булевых значений $(x_1, \ldots, x_n)$, что $F(x_1, \ldots, x_n) = 1$, то*

  $\mathrm{Prob}[$*число булевых наборов, на которых ψ обращается в единицу, нечетно*$] > 1 - 2^{-m}$,

- *если не существует такого набора булевых значений $(x_1, \ldots, x_n)$, что $F(x_1, \ldots, x_n) = 1$, то*

  $\mathrm{Prob}[$*число булевых наборов, на которых ψ обращается в единицу, нечетно*$] = 0$,

(Вероятность берется по случайным битам алгоритма.) При этом ψ является булевой комбинацией утверждений вида

$$F(y_1, \ldots, y_n) = 1 \ \&\ h(y_1, \ldots, y_n) = 0 \ldots 0$$

(для разных наборов $y_1, \ldots, y_n$ и разных аффинных отображений h). Алгоритм B работает за время $\mathrm{poly}(|\varphi|, m)$.

Доказательство состоит в буквальном повторении доказательства теоремы Теоремы 3. В самом деле, в этом доказательстве мы нигде не использовали внутренню структуру формулы φ, так что эту формулу можно заменить на утверждение $F(y_1, \ldots, y_n) = 1$ для произвольного предиката F.

Формулой класса Σ_k называют замкнутые формулы φ вида

$$\varphi = \exists a_1 \exists a_2 \ldots \forall b_1 \forall b_2 \ldots \exists c_1 \exists c_2 \ldots\ldots\ \theta(a_1, a_2, \ldots, b_2, b_2, \ldots, c_1, c_2, \ldots, \ldots),$$

где θ – некоторая булева формула, а в кванторной приставке кванторы чередуются не более k раз.

Будем обозначать QBF_k (булевы формулы с кванторами) множество всех истинных формул из Σ_k. Возникает естественная алгоритмическая задача – распознавание класса формул QBF_k. Другими словами, по формуле φ с кванторной приставкой с k чередованиями кванторов мы хотим выяснить, истинна эта формула или ложна.

Понятно, что данный класс алгоритмических задач некоторым образом обобщает уже привычные нам переборные задачи. Задача SAT является частным случаем задачи QBF_1. В самом деле, вопрос о выполнимости булевой формулы $\psi(x_1, \ldots, x_n)$ есть вопрос об истинности замкнутой формулы

$$\varphi = \exists x_1 \exists x_2 \ldots \exists x_n \theta(x_1, \ldots, x_n).$$

Теорема 4. *Для каждого k существует полиномиальный вероятностный алгоритм B_k, который по заданной формуле φ класса Σ_k строит такую булеву формулу ψ, что*

- *если φ истинная, то* $\mathrm{Prob}[\psi \in \oplus\mathrm{SAT}] > 1 - 2^{-m}$,
- *если φ ложна, то* $\mathrm{Prob}[\psi \in \oplus\mathrm{SAT}] < 2^{-m}$.

Доказательство[1] проведем индукцию по числу чередований кванторов k. Чтобы провести индукцию, нам потребуется доказывать немного более общее утверждение, чем условие теоремы.

Формулировка обобщенного варианта теоремы: Мы по-прежнему считаем, что формула

$$\varphi = \exists a_1 \exists a_2 \ldots \forall b_1 \forall b_2 \ldots \exists c_1 \exists c_2 \ldots\ldots \theta(a_1, \ldots, b_2, \ldots, c_2, \ldots, \ldots, z_1, \ldots, z_s),$$

имеет кванторную приставку с не более чем k чередованиями кванторов. Однако теперь мы разрешим формуле быть незамкнутой – она может содержать свободные переменные (параметры), не связанные никаким квантором. Мы покажем, что существует полиномиальный вероятностный алгоритм B_k, который по такой формуле φ строит булеву формулу

$$\psi(x_1, \ldots, x_l, z_1, \ldots, z_s)$$

(с $(l+m)$ переменными для некоторого $l \geq 0$) с некоторыми особыми свойствами. Заметим, что при подстановке в ψ вместо $(z_1, \ldots, z_s)$ набора булевых значений $(\zeta_1, \ldots, \zeta_s)$, мы получаем булеву формулу $\psi(x_1, \ldots, x_l, \zeta_1, \ldots, \zeta_s)$, в которой остается l переменных. Можно говорить о выполнимости этой формулы (l переменных, о числе её выполняющих наборов).

Мы потребуем, чтобы для любого набора значений параметров $(\zeta_1, \ldots, \zeta_s)$

- если $\varphi(\zeta_1, \ldots, \zeta_s)$ истинно, то

$$\mathrm{Prob}[\psi(x_1, \ldots, x_l, \zeta_1, \ldots, \zeta_s) \in \oplus\mathrm{SAT}] > 1 - 2^{-m},$$

- если $\varphi(\zeta_1, \ldots, \zeta_m)$ ложно, то

$$\mathrm{Prob}[\psi(x_1, \ldots, x_l, \zeta_1, \ldots, \zeta_s) \in \oplus\mathrm{SAT}] < 2^{-m}.$$

[1] См. также доказательство этой теоремы в [AB].

Итак, мы сформулировали «обобщенный» вариант теоремы (для формулы с параметрами). Далее мы докажем это утверждение индукцией по k (количеству чередований кванторов в кванторной приставке φ). База индукции ($k = 0$) очевидна – в качестве ψ можно взять саму формулу φ.

Теперь предположим, что обобщенный вариант теоремы доказан для $(k-1)$ перемен кванторов. Рассмотрим формулу φ вида

$$\exists a_1 \exists a_2 \ldots \forall b_1 \forall b_2 \ldots \exists c_1 \exists c_2 \ldots\ldots \theta(a_1, \ldots, b_2, \ldots, c_2, \ldots, \ldots, z_1, \ldots, z_s),$$

с k чередованиями кванторов (для определенности мы считаем, что самые внешние кванторы – это кванторы существования). Мы «отщепим» в этой формуле внешние кванторы существования и обозначим подформулу внутри этой кванторной приставки $\varphi'(a_1, a_2, \ldots, z_1, z_2, \ldots)$, т.е.,

$$\varphi(z_1, z_2, \ldots) = \exists a_1 \exists a_2 \ldots \varphi'(a_1, a_2, \ldots, z_1, z_2, \ldots).$$

Далее мы применим к $\varphi'(a_1, a_2, \ldots, z_1, z_2, \ldots)$ предположение индукции и получим такую формулу

$$\psi(a_1, a_2, \ldots, z_1, z_2, \ldots),$$

что для любого набора параметров $(\alpha_1, \alpha_2, \ldots, \zeta_1, \zeta_2, \ldots)$

- если $\varphi'(\alpha_1, \alpha_2, \ldots, \zeta_1, \zeta_2, \ldots)$ истинна, то

$$\mathrm{Prob}[\psi(\alpha_1, \ldots, \alpha_l, \zeta_1, \ldots, \zeta_s) \in \oplus\mathrm{SAT}] > 1 - 2^{-m-1},$$

- если $\varphi'(\alpha_1, \ldots, \alpha_l, \zeta_1, \zeta_2, \ldots)$ ложна, то

$$\mathrm{Prob}[\psi(\alpha_1, \ldots, \alpha_l, \zeta_1, \ldots, \zeta_s) \in \oplus\mathrm{SAT}] < 2^{-m-1}.$$

Теперь введем обозначение $F(x_1, \ldots, x_l, z_1, \ldots, z_s)$ для предиката

$$F(x_1, \ldots, x_l, z_1, \ldots, z_s) = \begin{cases} 1, & \text{если } \psi(\alpha_1, \ldots, \alpha_l, \zeta_1, \ldots, \zeta_s) \in \oplus\mathrm{SAT} \\ 0, & \text{иначе.} \end{cases}$$

Применим к формуле

$$\exists x_1 \ldots \exists x_l [F(x_1, \ldots, x_l, z_1, \ldots, z_s) = 1]$$

теорему $3'$ и построим выражение $\rho(z_1, \ldots, z_s)$ такое, что при любой подстановке булевых значений $\zeta_1, \ldots, z_s$ вместо переменных $z_1, \ldots, z_s$

- если существует такой набор булевых значений $(x_1, \ldots, x_l)$, что $F(x_1, \ldots, x_l) = 1$, то

  $\mathrm{Prob}[$число булевых наборов, на которых ρ обращается в единицу, нечетно$] > 1 - 2^{-m-1}$,

- если не существует такого набора булевых значений $(x_1, \ldots, x_l)$, что $F(x_1, \ldots, x_l) = 1$, то

  Prob[число булевых наборов, на которых ψ обращается в единицу, нечетно] $= 0$.

При этом построенная ψ будет булевой комбинацией выражений вида

$$F(y_1, \ldots, y_l) = 1 \ \& \ h(y_1, \ldots, y_l) = 0 \ldots 0$$

(для разных наборов $y_1, \ldots, y_n$ и разных аффинных отображений h)

Осталось сделать два последних замечания. Во-первых, булеву комбинацию утверждений вида

$$\psi(\alpha_1, \ldots, \alpha_l, \zeta_1, \ldots, \zeta_s) \in \oplus\text{SAT}$$

(и именно такая булева комбинация и вошла в итоговое выражение ρ) можно переписать как утверждение о (не)четности числа выполняющих наборов некоторой новой булевой формулы (см. замечание после Утверждения 1). И, во-вторых, сумма ошибок при применении шага индукции и при использовании теорему $3'$ даёт необходимую нам оценку $2^{-m-1} + 2^{-m-1} = 2^{-m}$. На этом шаг индукции заканчивается.

5 Лекция 5, 29 марта.

5.1 Теорема Тода.

Теорема 5 (Тода)**.** *Если существует полиномиальный (детерминированный или вероятностный) алгоритм для решения задачи* #SAT *(алгоритм, который по заданной булевой формуле подсчитывает число её выполняющих наборов), то для каждого k существует полиномиальный алгоритм (детерминированный или вероятностный) для задачи QBF_k.*

Доказательство: Для начала мы докажем следующую лемму:

Лемма 3. *Существует детерминированный полиномиальный алгоритм, который по заданному числу k и булевой формуле α строит такую формулу β, что*

- *если $\alpha \in \oplus$SAT, то $\#\beta = -1 \ mod \ 2^k$,*
- *если $\alpha \notin \oplus$SAT, то $\#\beta = 0 \ mod \ 2^k$.*

(Время работы алгоритма есть poly$(|\alpha|, k)$*).*

Доказательство леммы: Мы осуществим преобразование формулы α в формулу β в несколько этапов. Конструкция будет состоять из цепочки шагов

$$\alpha = \alpha_0 \to \alpha_1 \to \alpha_2 \ldots \to \alpha_{\log k} = \beta$$

При этом для каждой формулы α_i $(i = 1, 2, \ldots)$ будет выполняться условие:

- если $\#\alpha_{i-1} = -1 \ mod \ 2^{2^{i-1}}$, то $\#\alpha_i = -1 \ mod \ 2^{2^i}$,
- если $\#\alpha_{i-1} = 0 \ mod \ 2^{2^{i-1}}$, то $\#\alpha_i = 0 \ mod \ 2^{2^i}$

Чтобы этого добиться, достаточно гарантировать, что

$$\#\alpha_i = 4(\#\alpha_{i-1})^3 + 3(\#\alpha_{i-1})^4.$$

Требуемое преобразование можно организовать с помощью конструкции из Утверждения 1 из лекции 4. Повторив данное преобразование $O(\log k)$ раз, мы получим нужную нам формулу β. Заметим, что размер формулы β ограничен полиномом от размера α и числа k. Лемма доказана.

Вернемся к доказательству теоремы. В Теореме 4 из лекции 4 мы построили алгоритм B, который по заданной замкнутой формуле φ (булевой формуле с кванторами с c чередованиями кванторов) и набору битов случайных $r_1 \dots r_l$ строит некоторую булеву формулу $\alpha(x_1, \dots, x_n, y_1, \dots, y_l)$ с $n + l$ переменными, для которой выполняются условия:

- если φ истинна, то $\text{Prob}_{r_1 \dots r_l}[\alpha(x_1, \dots, x_n)|_{y_1 = r_1 \dots y_l = r_l} \in \oplus\text{SAT}] > 1 - 2^{-m}$,
- если φ ложна, то $\text{Prob}_{r_1 \dots r_l}[\alpha(x_1, \dots, x_n)|_{y_1 = r_1 \dots y_l = r_l} \in \oplus\text{SAT}] < 2^{-m}$.

Т.е., при подстановке в α вместо переменных $y_1, \dots, y_l$ случайных значений $r_1 \dots r_l$ мы с большой вероятностью получаем булеву формулу, в которой остается n переменных и которая имеет «правильную» чётность числа выполняющих наборов.

Далее мы применим к формуле α преобразование из Леммы 3 для $k = l + 1$ и получим некоторую формулу $\beta(z_1, \dots, z_m, y_1, \dots, y_l)$.

Мы можем считать, что при этом $y_1, \dots, y_l$ остаются свободными переменными новой формулы β. Можно сказать иначе: каждому набору случайных битов $r_1 \dots r_l$ соответствует своя формула $\beta(z_1, \dots, z_m)$; но при этом общая формула $\beta(z_1, \dots, z_m, y_1, \dots, y_l)$ может быть выписана явно. Что можно сказать про число выполняющих наборов построенной формулы β? Рассмотрим два возможных случая.

Случай первый: Пусть исходная формула φ была истинной. Тогда при подстановке $\geq \frac{3}{4} \cdot 2^l$ возможных значений на место переменных $y_1, \dots, y_l$ мы будем получать формулу $\beta(z_1, \dots, z_m)$, для которой $\#\beta(z_1, \dots, z_m) = -1 \ mod \ 2^{l+1}$. Следовательно,

$$\#\beta(z_1, \dots, z_m, y_1, \dots, y_l) \text{ лежит в интервале } [-2^l, -(3/4) \cdot 2^l] \ (mod \ 2^{l+1}),$$

Второй первый: Пусть исходная формула φ была ложной. Тогда при подстановке $\geq \frac{3}{4} \cdot 2^l$ возможных значений на место переменных $y_1, \dots, y_l$ мы будем получать формулу $\beta(z_1, \dots, z_m)$, для которой $\#\beta(z_1, \dots, z_m) = 0 \ mod \ 2^{l+1}$. Следовательно,

$$\#\beta(z_1, \dots, z_m, y_1, \dots, y_l) \text{ лежит в интервале } [-(1/4) \cdot 2^l, 0] \ (mod \ 2^{l+1}).$$

Интервалы значений $\#\beta$ в первом и втором случае не пересекаются. Таким образом, если мы можем узнать число выполняющих наборов для β, то мы в результате узнаем, истинна или ложна исходная формула φ. Теорема доказана.

(См. также доказательство теоремы Тода в [AB])

5.2 Интерактивные доказательства.

Определение классов IP (интерактивные доказательства) и MA (игры Артура и Мерлина), интерактивный протокол для задачи неизоморфизм графов (с секретными случайными битами верифайера), см. [Si, AB]

6 Лекция 6, 5 апреля.

Игра Артура и Мерлина для задачи неизоморфизм графов (случайные биты Артура доступны Мерлину).

Игра Артура и Мерлина для той же задачи с односторонней ошибкой: если графы неизоморфны, то Артур с вероятностью 1 выдает ответ 1 (при правильном поведении Мерлина); если графы неизоморфны, то Артур с вероятностью не менее 3/4 выдает ответ нет (при любой стратегии Мерлина).

7 Лекция 7, 12 апреля (прочитана А. Шенем).

Содержание лекции:

1. Доказательство $\mathsf{IP}[poly] = \mathsf{AM}[poly]$ методом Килиана: протокол интерактивного доказательства с секретными случайными битами Верифайера можно преобразовать в протокол игры Артура и Мерлина с открытыми случайными битами Артура (число раундов увеличивается, но остается полиномиальным).

2. Определения доказательства с нулевым разглашением. [Va, Go]

3. Доказательство с нулевым разглашением для задачи *изоморфизм графов*. [AB, Va, Go]

8 Лекция 8, 26 апреля: ветвящиеся программы и теорема Баррингтона.

В этой лекции мы рассмотрим *ветвящиеся программы* (*branching program*). Ветвящиеся программы изучаются как своеобразной моделью вычислений с ограниченной памятью.

Ветвящаяся программа для вычисления функции $f : \{0,1\}^n \to \{0,1\}$ определяется как ориентированный ациклический граф, вершины которого разделены на слои $0, 1, \ldots, L$, причём

- из вершин слоя i рёбра могут вести только в вершины слоя $i+1$,
- в слое номер 0 имеется единственная вершина, её входная степень равна нулю,
- каждая вершина слоя L помечена нулём или единицей; выходные степени вершин слоя L равны нулю;
- каждая вершина слоёв $0, \ldots, L-1$ помечена одной из переменных $\{x_1, \ldots, x_n\}$, и имеет два выходящих ребра с пометками 0 и 1.

Процедура вычисления функции $f(x_1, \ldots, x_n)$ с помощью такой программы состоит в том, что мы проходим от единственной вершины слоя 0 до некоторой вершины слоя L по ориентированным рёбрам. Проходя через вершину, помеченную переменной x_i, мы считываем значение аргумента x_i и далее двигаемся по выходящему ребру с соответствующей пометкой. Добравшись до вершины уровня L, мы узнаем значение функции $f(x_1, \ldots, x_n)$ из пометки на финальной вершине.

Число уровней L иногда называют *длиной* ветвящейся программы. *Шириной* W ветвящейся программы называется максимальное число вершин, находящихся на одном уровне.

Несколько неформально можно сказать, что число уровней L соответствует времени выполнения программы, а ширина W есть память, используемая программой.

Упражнение. Докажите, что любая функция $f : \{0,1\}^n \to \{0,1\}$ вычисляется некоторой ветвящейся программой с длиной $O(n2^n)$ и шириной 3.

Теорема 6 (Баррингтон)**.** *Если для функции $f : \{0,1\}^n \to \{0,1\}$ существует булева схема глубины d, то f можно вычислить ветвящейся программой ширины 5 и длины $L \leq 4^d$.*

Из теоремы Баррингтона немедленно вытекает следующее следствие:

Следствие 1. *Если функция f вычислима булевой схемой логарифмической глубины, то эта функция вычислима ветвящейся программой полиномиальной длины ширины 5.*

Упражнение. Докажите, что функции *чётность*

$$\mathsf{Parity}(x_1, \ldots, x_n) = x_1 \oplus \ldots \oplus x_n$$

и *большинство*

$$\mathsf{Majority}(x_1, \ldots, x_n) = \begin{cases} 1, & \text{если } x_1 + \ldots + x_n \geq n/2, \\ 0, & \text{иначе.} \end{cases}$$

можно вычислить булевыми схемами глубины $O(\log n)$.

Доказательство теоремы Баррингтона: Мы предполагаем, что булевы схемы могут содержать функциональные элементы *конъюнкция* и *дизъюнкция* с входной степенью 2 и *отрицание* с входной степенью 1.

Мы должны научиться для каждой булевой схемы глубины d строить ветвящуюся программу ширины 5 и длины не более 4^d. Далее мы покажем, как преобразовать произвольную булеву схему в ветвящуюся программу. При этом полученная программа будет обладать дополнительным свойством: на каждом уровне $i = 0, \ldots, L$ все вершины будут помечены одной и той же переменной $x_{j(i)}$.

Прежде чем доказывать нужное нам утверждение, полезно ввести понятие *групповой ветвящейся программы* (или G-программы). Пусть имеется некоторая группа G с нейтральным элементом $e \in G$. Тогда G-программой длины L называется пара последовательностей элементов данной группы $(g_1^0, \ldots, g_L^0)$, $(g_1^1, \ldots, g_L^1)$ вместе с набором натуральных чисел $(k_1, \ldots, k_L)$. Будем говорить, что данная G-программа α-вычисляет функцию $f : \{0,1\}^n \to \{0,1\}$, если

$$f(x_1, \ldots, x_n) = 1 \;\Rightarrow\; \prod_{i=1}^{L} g_i^{x_{k_i}} = \alpha$$

и

$$f(x_1, \ldots, x_n) = 0 \;\Rightarrow\; \prod_{i=1}^{L} g_i^{x_{k_i}} = e.$$

Более единообразно данное свойство можно переписать в виде

$$\prod_{i=1}^{L} g_i^{x_{k_i}} = \alpha^{f(x_1,\ldots,x_n)}.$$

Нам будут полезны ветвящиеся групповые программы над группой S_5 (группой перестановок 5 элементов). Далее мы покажем, что булеву схему глубины d можно переделать в S_5-программу длины 4^d; затем (это уже совсем просто) мы превратим S_5-программу в ветвящуюся программу ширины 5.

Лемма 4 (все циклы длины 5 эквивалентны)**.** *Пусть α и β есть два цикла длины 5 из группы S_5 и $f : \{0,1\}^n \to \{0,1\}$ некоторая булева функция. Тогда, если функция f α-вычисляется S_5-программой длины L, то f также β-вычисляется некоторой S_5-программой такой же длины.*

Доказательство леммы: Содержательно утверждение этой леммы очевидно: все циклы длины 5 «устроены» одинаково и отличаются друг от друга только переименованием элементов $\{1, \ldots, n\}$. Для формального доказательства нужно заметить, что α и β являются сопряженными элементами в S_5, т.е., существует такая перестановка $\pi \in S_5$, что $\beta = \pi^{-1} \cdot \alpha \cdot \pi$.

Далее, пусть наборы перестановок $(g_1^0, g_2^0 \ldots, g_{L-1}^0, g_L^0)$, $(g_1^1, g_2^1 \ldots, g_{L-1}^1, g_L^1)$ вместе с набором натуральных чисел $(k_1, \ldots, k_L)$ α-вычисляют f. Тогда наборы перестановок $(\pi g_1^0, g_2^0 \ldots, g_{L-1}^0, g_L^0 \pi^{-1})$, $(\pi g_1^1, g_2^1 \ldots, g_{L-1}^1, g_L^1 \pi^{-1})$ вместе

с тем же набором чисел $(k_1, \ldots, k_L)$ β-вычисляют данную функцию f. В самом деле, если

$$\prod_{i=1}^{L} g_i^{x_{k_i}} = e,$$

то

$$\pi^{-1} \cdot \prod_{i=1}^{L} g_i^{x_{k_i}} \cdot \pi = \pi^{-1} \cdot e \cdot \pi = e.$$

Аналогично, если

$$\prod_{i=1}^{L} g_i^{x_{k_i}} = \alpha,$$

то

$$\pi^{-1} \cdot \prod_{i=1}^{L} g_i^{x_{k_i}} \cdot \pi = \pi^{-1} \cdot \alpha \cdot \pi = \beta.$$

Лемма доказана.

Лемма 5 (от программы для f к программе для $\neg f$)**.** *Если функция $f : \{0,1\}^n \to \{0,1\}$ α-вычисляется некоторой S_5-программой длины L, то и отрицание этой функции $\neg f$ тоже вычисляется S_5-программой длины L.*

Доказательство леммы: Лемма 4 гарантирует, что существует S_5-программа длины L, которая α^{-1}-вычисляет f. Умножим последние элементы этой программы (g_L^0 и g_L^1) на α и получим S_5-программу для вычисления $\neg f$.

Лемма 6 (от программ для f и g к программе для конъюнкции $(f\&g)$)**.** *Если функция $f : \{0,1\}^n \to \{0,1\}$ α-вычисляется некоторой S_5-программой длины L и функция $g : \{0,1\}^n \to \{0,1\}$ β-вычисляется некоторой S_5-программой длины L, то конъюнкция $f\&g$ является $(\alpha\beta\alpha^{-1}\beta^{-1})$-вычислимой с помощью некоторой программы длины $4L$.*

Доказательство леммы: Чтобы получить нужную нам программу, достаточно взять конкатенацию четырёх уже имеющихся программ:

- α-вычисление f,
- β-вычисление g,
- α^{-1}-вычисление f,
- β^{-1}-вычисление g.

Нетрудно проверить, что данная вычисляет конъюнкцию f и g в нужном нам смысле:

- если для некоторого набора значений переменных выполнено $f(x_1, \ldots, x_n) = 1$ и $g(x_1, \ldots, x_n) = 1$, то указанная конкатенация программ возвратит значение $\alpha\beta\alpha^{-1}\beta^{-1}$,

- если $f(x_1, \ldots, x_n) = 0$ и $g(x_1, \ldots, x_n) = 1$, то данная конкатенация программ возвратит значение $e \cdot \beta \cdot e \cdot \beta^{-1} = e$,
- если $f(x_1, \ldots, x_n) = 1$ и $g(x_1, \ldots, x_n) = 0$, то данная конкатенация программ возвратит значение $\alpha \cdot e \cdot \alpha^{-1} \cdot e = e$,
- если $f(x_1, \ldots, x_n) = 0$ и $g(x_1, \ldots, x_n) = 0$, то данная конкатенация программ возвратит значение $e \cdot e \cdot e \cdot e = e$.

Лемма доказана.

Напомним, что в группе S_5 для некоторых циклов α и β (длины 5) их коммутатор $\alpha\beta\alpha^{-1}\beta^{-1}$ тоже является циклом длины 5. Это верно, например, для циклов (12345) и (13542).

Замечание: По существу, мы воспользовались тем фактом, что группа S_5 неразрешима.

Теперь мы можем индуктивно строить S_5-программу, соответствующую заданной схеме из функциональных элементов. При этом полученная программа будет иметь длину не более 4^d, где d есть глубина исходной схемы. В самом деле, добавление к булевой схеме отрицания не увеличивает сложности соответствующей программы, см. Лемму 5. А при соединении двух схем в одну с помощью конъюнкции требуется последовательно соединить по две копии программы для каждой их двух подсхем, см. Лемму 6.

Для завершения доказательства теоремы Баррингтона нам остается убедиться в справедливости следующей леммы.

Лемма 7. *Если некоторая булева функция α-вычислима S_5-программой длины L для некоторого цикла α (длины 5), то эта функция вычислима также некоторой ветвящейся программой ширины 5 и длины L.*

Доказательство леммы: Пусть S_5-программа для вычисления функции f задана наборами перестановок $(g_1^0, g_2^0 \ldots, g_{L-1}^0, g_L^0)$ и $(g_1^1, g_2^1 \ldots, g_{L-1}^1, g_L^1)$ и набором натуральных чисел $(k_1, \ldots, k_L)$. Согласно Лемме 4 можно без ограничения общности предполагать, что $\alpha = (12345)$.

Рассмотрим ветвящуюся программу, в которой в каждом слое $i = 1, \ldots, L$ имеется по 5 элементов (занумерованных числами от 1 до 5), и рёбра из вершин уровня i в вершины уровня $i+1$ соответствуют перестановкам g_i^0 и g_i^1. Стартовой вершиной (на уровне номер 0) будем считать вершину 1; на последнем уровне (уровне номер L) пометим единицей вершину 2 и нулём вершину 1 (вершины $3, 4, 5$ уровня L можно пометить произвольным образом).

Если $f(x_1, \ldots, x_n) = 1$, то по определению S_5-программы

$$\prod_{i=1}^{L} g_i^{x_{k_i}} = (12345),$$

а значит в построенной нами ветвящейся программе из стартовой вершины 1 нулевого уровня мы попадаем финальную в вершину 2 последнего уровня.

Если же $f(x_1, \ldots, x_n) = 0$ и (снова по определению S_5-программы)

$$\prod_{i=1}^{L} g_i^{x_{k_i}} = e,$$

то в построенной нами ветвящейся программе из стартовой вершины 1 нулевого уровня мы попадаем финальную в вершину 1. Тем самым, доказательство завершено.

9 Лекция 9, 10 мая.

Недетерминированные вычисления. Два эквивалентных определения класса NP. Сводимость по Карпу, понятие NP-полнота. Теорема Кука–Левина: NP-полнота задач выполнимость и 3-КНФ. [Si, AB]

10 Лекция 10, 17 мая.

Самосводимость языка SAT. Теорема Махейни (Mahaney).

Определение 1. *Множество $S \subset \{0,1\}^*$ будем называть* тощим, *если все слова множества S состоят из одних лишь единиц. (Другими словами, $S \subset 1^*$.)*

Определение 2. *Множество $S \subset \{0,1\}^*$ называется* разреженным *(sparse), если $|S \cap \{0,1\}^n| \leq \mathsf{poly}(n)$.*

Теорема 7. *Если существует* NP*-полное тощее множество, то* P = NP.

Доказательство: Пусть множество A является тощим и при этом NP-полное. В таком случае к задаче распознавания A сводится задача SAT (задача выполнимости), т.е., существует вычислимая за полиномиальное время функция f, такая что для всякой булевой формулы φ

$$\varphi \in \mathrm{SAT} \Leftrightarrow f(\varphi) \in A.$$

Как мы увидим ниже, факт существования данной сводимости f оказывается очень полезен, даже если у нас нет средств для распознавания языка A.

Прежде всего заметим, что вопрос о выполнимости булевой формулы с n переменными $\varphi(x_1, \ldots, x_n)$ очевидным образом сводится к вопросу о выполнимости двух булевых формул с $n-1$ переменной:

$$\varphi(x_1, x_2, \ldots, x_n) \in \mathrm{SAT} \Leftrightarrow (\varphi(0, x_2 \ldots, x_n) \in \mathrm{SAT} \text{ или } \varphi(1, x_2 \ldots, x_n) \in \mathrm{SAT}).$$

Это простое свойство языка SAT часто называют *самосводимостью.*

Разумеется, описанную «самосводимость» можно применить несколько раз:

$$\begin{aligned}
\varphi(x_1, x_2, \ldots, x_n) \in \mathrm{SAT} \quad &\Leftrightarrow \quad \varphi(0, x_2 \ldots, x_n) \in \mathrm{SAT} \text{ или } \varphi(1, x_2 \ldots, x_n) \in \mathrm{SAT} \\
&\Leftrightarrow \quad \varphi(0, 0, x_2 \ldots, x_n) \in \mathrm{SAT} \text{ или } \varphi(0, 1 \ldots, x_n) \in \mathrm{SAT} \\
&\qquad \text{или } \varphi(1, 0 \ldots, x_n) \in \mathrm{SAT} \text{ или } \varphi(1, 1 \ldots, x_n) \in \mathrm{SAT} \\
&\Leftrightarrow \quad \ldots
\end{aligned}$$

На i-ом шаге такой редукции мы сводим исходный вопрос о выполнимости формулы с n переменными $\varphi(x_1, \ldots, x_n)$ к вопросу о выполнимости 2^i формул с $(n-i)$ переменными. Через n шагов этого процесса мы сведем исходный вопрос к вопросу об истинности булевых формул, вовсе не имеющим переменных (т.е., константам), и таким образом узнаем, выполнима ли исходная формула φ.

Однако у данной тривиальной процедуры есть очевидный изъян: в процессе самосводимости список рассматриваемых формул экспоненциально растет. Мы же хотим построить *полиномиальную* процедуру для проверки выполнимости φ.

Чтобы сократить список рассматриваемых формул, мы добавим к описанной процедуре дополнительную проверку, которая позволит исключать из рассмотрения «лишние» формулы. Для каждого шага самосводимости для всех добавляемых в список формул ψ (полученных из $\varphi(x_1, \ldots, x_n)$ подстановкой некоторых булевых значений вместо части переменных) мы будем вычислять $f(\psi)$ (это можно сделать за полиномиальное время). Если $f(\psi)$ не состоит из одних только единиц, то $f(\psi)$ заведомо не может принадлежать A, а значит данная формула ψ заведомо не является выполнимой. Такую формулу можно исключить из дальнейшего рассмотрения. Отметим, что если $f(\psi) \in 1^*$, то мы не можем заключить, выполнима ли такая формула (у нас пока нет средств для распознавания языка A за полиномиальное время).

Заметим, что если для каких-то двух формул из нашего списка значения сводящей функции f оказываются одинаковыми ($f(\psi_1) = f(\psi_2)$), то эти формулы либо одновременно выполнимы, либо одновременно не выполнимы. Это значит, что достаточно добавить вы список только одну из этих формул.

Таким образом, в процессе редукции (мы шаг за шагом сводим вопрос о выполнимости исходной формулы $\varphi(x_1, \ldots, x_n)$ к вопросам о выполнимости формул с меньшим числом переменных) на каждом шаге i достаточно оставлять в списке формул с $(n-i)$ переменными только формулы, у каждой из которых f-значение состоит из одних единиц; при этом для каждого образа функции f (состоящего из одних единиц) достаточно оставить только одну такую формулу.

Поскольку f вычислима за полиномальное время, на каждом шаге нашей редукции размер списка формул, требующих нашего внимания, не будет превосходить размера $\mathrm{poly}(|\varphi|)$.

Ясно, что с использованием описанного «подрезания» списка формул

всю процедура сведения φ к дизъюнкции булевых констант (формул без переменных) можно провести за полиномиальное время.

Таким образом, мы получили полиномиальный алгоритм для решения NP-полной задачи SAT. Тем самым, в предположении о существовании тощего NP-полного множества мы получили P = NP. Теорема доказана.

Теорема 8. *Если существует* coNP*-полное разреженное множество, то* P = NP*.*

Доказательство: этой теоремы мы оставляем в качестве упражнения.

Теорема 9 (Mahaney)**.** *Если существует* NP*-полное разреженное множество, то* P = NP*.*

Доказательство: Предположим, что среди NP-полных множеств есть разреженное множество A. Это значит, что любое множество из класса NP сводится по Карпу к множеству A. Мы покажем, что из этого предположения следует P = NP.

В доказательстве Теоремы 7 мы воспользовались сводимостью к разреженному множеству A стандартной NP-полной задачи SAT. А в данном доказательстве нам потребуется не вполне обычная задачу из класса NP, которую мы будем называть SAT-hint.

Определим язык SAT-hint следующим образом:

$$\text{SAT-hint} = \left\{ (\varphi, \rho) \ : \begin{array}{l} \varphi \text{ есть булева формула } n \text{ переменных, } \rho \text{ есть набор из } k \le n \text{ нулей и единиц,} \\ \text{и для формулы } \varphi \text{ существует выполняющий набор из } n \text{ булевых значений } \rho' \\ \text{лексикографически больший или равный } \rho \end{array} \right\}$$

Замечание: Полезно представлять себе все булевы наборы ρ в виде вершин двоичного дерева высоты n. Если рисовать дерево растущим сверху вниз, то $\rho_1 < \rho_2$ (ρ_1 лексикографически меньше ρ_2) означает, что ρ_2 есть продолжение ρ_1 или вершина ρ_2 расположена в дереве правее вершины ρ_1.

Отметим, что если у булевой формулы $\varphi(x_1, \ldots, x_n)$ есть хотя бы один выполняющий набор, то пары (φ, Λ) и $(\varphi, 0 \ldots 0)$ принадлежат SAT-hint (пустое слово Λ и слово из одних нулей $0 \ldots 0$ лексикографически меньше или равны любого набора из n булевых значений, а значит меньше или равны выполняющего набора формулы φ).

Очевидно, задача SAT-hint лежит в классе NP. (На самом деле язык SAT-hint не просто лежит в NP, а является NP-полной задачей; но это свойство нам в доказательстве не понадобится.) Следовательно, SAT-hint сводится по Карпу к множеству A, т.е., существует вычислимая за полиномиальное время функция f такая, что

$$(\varphi, \rho) \in \text{SAT-hint} \Leftrightarrow f(\varphi, \rho) \in A.$$

С помощью этой функции f мы построим полиномиальный алгоритм для решения задачи SAT (здесь речь идет уже об обычной задаче SAT, а не SAT-hint). Тем самым, в предположениях теоремы мы получим заключение P = NP.

Итак, пусть дана некоторая булева формула n переменных $\varphi(x_1, \ldots, x_n)$. Мы хотим узнать, есть ли для этой формулы выполняющий набор значений $(x_1, \ldots, x_n)$. Покажем, как (с помощью определенной выше функции f) найти ответ на этот вопрос за полиномиальное время. Более точно, мы покажем, как найти лексикографически максимальный выполняющий набор для формулы φ (если хотя бы один выполняющий набор существует).

Напомним, что пустое слово предшествует в лексикографическом порядке всем остальным словам. Поэтому булева формула $\varphi(x_1, \ldots, x_n)$ выполнима, если и только если

$$(\varphi, \Lambda) \in \text{SAT-hint}.$$

Далее мы воспользуемся свойством самосводимости:

$$\begin{aligned}(\varphi, \Lambda) \in \text{SAT-hint} \quad &\Leftrightarrow \quad (\varphi, 0) \in \text{SAT-hint или } (\varphi, 1) \in \text{SAT-hint} \\ &\Leftrightarrow \quad (\varphi, 00) \in \text{SAT-hint или } (\varphi, 00) \in \text{SAT-hint} \\ &\qquad\quad \text{или } (\varphi, 10) \in \text{SAT-hint или } (\varphi, 11) \in \text{SAT-hint} \\ &\Leftrightarrow \quad \ldots\end{aligned}$$

На каждом следующем шаге редукции мы сводим исходный вопрос про (φ, Λ) к дизъюнкции 2^i вопросов (φ, ρ) со всевозможными наборами ρ, состоящими из i булевых значений. Когда длина ρ дорастет до n, мы сможем подставить каждый такой набор булевых значений в φ и вычислить соответствующее булево значение. В результате мы узнаем, есть ли у формулы φ выполняющий набор (и даже найдем лексикографически максимальный среди всех выполняющих наборов).

Однако данный план невозможно реализовать наивным образом: с ростом i число булевых наборов ρ увеличивается экспоненциально. Нам потребуется применить некоторый трюк, который позволит ограничить размер списка обрабатываемых ρ полиномом. Для этого нам и понадобится сводящая функция f.

Функция f вычисляется за полиномиальное время, так что для всех наборов ρ (длины не более n) значение $f(\varphi, \rho)$ ограничено некоторым полиномом от длины формулы φ. Напомним, что множество A является разреженным. Это значит, что число всевозможных элементов A, в с которыми может совпасть значение $f(\varphi, \rho)$, также не превосходит некоторого числа $N = \text{poly}(|\varphi|)$. Далее мы покажем, что при поиске выполняющего набора для φ размер списка «потенциально интересных» наборов ρ можно ограничить именно этим числом N.

На каждом шаге i мы будем действовать следующим образом. Мы возьмем все булевы наборы ρ, появившиеся на $(i-1)$-ом шаге, и двумя способами продолжим каждый из них до набора с длиной на единицу больше (каждый набор ρ продолжается до $\rho 0$ и $\rho 1$). Таким образом, число рассматриваемых наборов увеличивается в два раза. Далее мы применим к каждому из получившихся наборов функцию f. Если для каких-то двух наборов ρ_1 и ρ_2 соответствующие f-значения окажутся одинаковыми ($f(\varphi, \rho_1) = f(\varphi, \rho_2)$), то утверждения $(\varphi, \rho_1) \in \text{SAT-hint}$ и $(\varphi, \rho_2) \in \text{SAT-hint}$ эквивалентны; следовательно, один из двух наборов ρ_1, ρ_2 можно исключить из дальнейшего

рассмотрения. Если ρ_1 меньше ρ_2, то лексикографически максимальный выполняющий набор заведомо не может встретиться среди продолжений ρ_1; так что мы выбросим из рассмотрения ρ_1.

Таким образом, мы исключим из дальнейшего рассмотрения часть полученных булевых наборов (для оставшихся в рассмотрении наборов ρ_i все значения $f(\varphi, \rho_i)$ будут различными). Если в результате получился список из не более, чем N наборов (напомним, что $N = \mathrm{poly}(|\varphi|)$), то можно перейти к следующему этапу и увеличить число i ещё на единицу. Если же размер списка оказался больше N, то можно исключить из рассмотрения «хвост» этого списка и оставить в нем лишь только N первых (в лексикографическом порядке) наборов. При этом мы заведомо не исключим из рассмотрения набор ρ_i, продолжение которого является самым правым (лексикографически самым большим) выполняющим набором формулы φ.

В самом деле, если $(\varphi, \rho) \in$ SAT-hint для некоторого набора из i булевых значений ρ, то мы имеем (по определению SAT-hint) $(\varphi, \rho') \in$ SAT-hint и для всех ρ', лексикографически *меньших* данного ρ. Таким образом, если среди выписанных в лексикографическом порядке булевых наборов

$$\rho_1, \rho_2, \ldots, \rho_s$$

найдется набор ρ_i, для которого $(\varphi, \rho_i) \in$ SAT-hint, то и все предшествующие пары $(\varphi, \rho_1), \ldots, (\varphi, \rho_{i-1})$ тоже принадлежат SAT-hint. В то же время, для наборов $\rho_1 \ldots, \rho_{N+1}$ все значения $f(\varphi, \rho_i)$ различны (это значит, что *не все* эти значения попадают в A). Следовательно, среди этих $N+1$ пар не все принадлежат SAT-hint. Но это и означает, что максимальный ρ_i, для которого $(\varphi, \rho_i) \in$ SAT-hint, может находится лишь среди первых N наборов. Следовательно, мы имеем право исключить из дальнейшего рассмотрения наборы $\rho_{N+1}, \rho_{N+2}, \ldots, \rho_s$ (и их продолжения).

Таким образом, мы получили алгоритм, который находит выполняющий набор произвольной выполнимой булевой формулы за полиномиальное время. Теорема Махейни доказана.

11 Лекция 11, 25 мая.

Содержание лекции:

1. Теоремы об иерархии по времени и памяти (для детерминированных вычислений). [AB, Si]
2. Вычисление с логарифмической памятью. Самосводимость задачи достижимости на ориентированном графе (достижима ли вершина t из вершины s за $\leq k$ шагов).
3. Теорема Сэвича: $\mathsf{Space}(f(n)) \subset \mathsf{Space}(f^2(n))$.
   Следствие: $\mathsf{NPSPACE} \subset \mathsf{NPSPACE}$. [AB]
4. Теорема Фортноу: Не существует алгоритма, решающего задачу SAT за время $n^{1+o(n)}$ с использованием памяти $O(\log n)$.

Список литературы

[AB] Sanjeev Arora and Boaz Barak. Computational Complexity: A Modern Approach.

[Si] Michael Sipser. Introduction to the theory of computation.

[MVV] Ketan Mulmuley, Umesh V. Vazirani. Vijay V. Vazirani. Matching as Easy as Matrix Inversion. Combinatorica, Volume 7, Issue 1, pp. 105–113, 1987.

[Va] S. P. Vadhan. A Study of Statistical Zero-Knowledge Proofs. PhD thesis, Massachusetts Institute of Technology, 1999.

[Go] O. Goldreich. Foundations of Cryptography, Volumes 1 and 2. Cambridge University Press, 2001, 2004.

[HO] Lane A. Hemasphaandra, Mitsunori Ogihara. The Complexity Companion. Springer, 2002.