Лекция 11

Введение в сложность в среднем

(Конспект: Г. Ярославцев)

Как мы знаем из первой лекции, односторонние в наихудшем случае функции существуют тогда и только тогда, когда $P \neq NP$. Хотя односторонние в наихудшем случае функции практического интереса не представляют, у них очень привычное для теории сложности определение, задача их обращения не решается за полиномиальное время. Криптографически односторонние функции не очень хороши с точки зрения теории сложности, поскольку их существование не следует ни из каких разумных предположениях о сложностных классах. Важнейшим открытым вопросом является нахождение таких предположений. Разумно искать такие предположения на языке не сложности в наихудшем случае, а на языке сложности в среднем случае.

Теория сложности в среднем случае возникла в 80-х годах 20-го века, ее основоположниками стали Л. Левин и Ю. Гуревич.

11.0.1 Определения

В теории сложности в среднем все вычислительные задачи рассматриваются вместе с вероятностными распределениями на исходных данных.

Определение 11.1 (Ансамбль распределений). Мы будем говорить, что имеется ансамбль распределений $\{\mathcal{D}_n\}_{n=1}^{\infty}$, если задана последовательность распределний $\mathcal{D}_i : \{0,1\}^* \to [0,1]$ и существует такой многочлен $p(n)$, что

$$\sum_{x \in \{0,1\}^*} \mathcal{D}_n(x) = \sum_{|x| \leq p(n)} \mathcal{D}_n(x) = 1$$

. Здесь n — это характерный размер входа для задачи (например, число вершин в графе, число переменной в формуле или просто размер входа). Носителем распределения называется множество $supp\,\mathcal{D}_n = \{x \in \{0,1\}^n \mid \mathcal{D}_n(x) > 0\}$.

Пример 11.1. $U_n(x) = 2^{-n}$, если $|x| = n$. Это ансамбль равномерных распределений.

Определение 11.2. Распределенной задачей распознавания называется пара $(L, \mathcal{D})$, где $L \subseteq \{0,1\}^*$ — это язык, а $\mathcal{D}$ это ансамбль распределений.

Нашей ближайшей целью является определение понятия *алгоритм решает задачу за полиномиальное в среднем время*. Начнем с наивного определения. Будем говорить, что распределенная задача распознавания $(L, \mathcal{D})$ решается за полиномиальное в среднем время, если существует такой алгоритм A, что он распознает язык L и математическое ожидание времени его работы на входах, з адаваемых ансамблем распределением $\mathcal{D}_n$, ограничено некоторым полиномом от n.

Почему данное выше определение нельзя считать хорошим? Дело в том, что класс определенных выше задач не замкнут относительно такой п ростой операции как возведение в квадрат, например (напомним, что класс полиномов как раз тем и хорош, что замкнут относительно сложения и умножения). Действительно, пусть $\mathcal{D}_n$ это равномерное распределение на входах длины n, A — алгоритм, распознающий язык L, а $t_A(x)$ — время работы алгоритма A на входе x. Пусть $t_A(x) = 2^n$, если $x = 0^n$ и $t_A(x) = 1$ при $x \neq 0^n$. В таком случае $E[t_A(x)]$ ограничено полиномом, а $E[t_A^2(x)]$ уже экспоненциально.

Приведенный пример показывает, что наивное определение не очень удачно. Поэтому наше определение будет чуть более слабым.

Определение 11.3 (Левин). Алгоритм A решает распределенную задачу распознавания $(L, \mathcal{D})$ за полиномиальное в среднем время, если $\exists \epsilon > 0 : \underset{x \leftarrow \mathcal{D}_n}{E}[t_A^\epsilon(x, 1^n)] = O(n)$.

Другое эквивалентное определение дал Импальяццо.

Определение 11.4 (Импальяццо). Задача распознавания $(L, \mathcal{D})$ решается за полиномиальное в среднем время, если существует алгоритм $A(x, \delta, 1^n)$ (здесь δ это рациональное число из интервала [0..1]) такой, что:

1. Время работы $A(x, \delta, 1^n)$ ограничено некоторым полиномом $p(\frac{|x|}{\delta})$
2. $A(x, \delta, 1^n) \in \{L(x), \perp\}$ при $x \in supp\, \mathcal{D}_n$
3. $\underset{x \leftarrow \mathcal{D}_n}{\Pr}[A(x, \delta, 1^n) = \perp] < \delta$

Теорема 11.1. *Определения Левина и Импальяццо эквивалентны.*

Доказательство. Покажем, что из определения Левина следует определение Импальяццо.

Пусть задача распознавания $(L, \mathcal{D})$ решается за полиномиальное в среднем время в определении Левина. Тогда существуют такой алгоритм A и константы ϵ, c, что $\underset{x \leftarrow \mathcal{D}_n}{E}[t_A(x, 1^n)^\epsilon] \leq cn$. В таком случае мы можем построить алгоритм $B(x, \delta, 1^n)$, который удовлетворяет определению Импальяццо. Алгоритм B запускает $A(x, 1^n)$ на $(\frac{c|x|}{\delta})^{\frac{1}{\epsilon}}$ шагов. Если успевает остановиться, то B выдает его ответ, иначе выдает $\perp$.

Свойства 1 и 2 из определения Импальяццо выполнены, осталось проверить выполнение Свойства 3:

$$\underset{x \leftarrow \mathcal{D}_n}{\Pr}[B(x, \delta, 1^n) = \perp] = \underset{x \leftarrow \mathcal{D}_n}{\Pr}[t_A(x, 1^n) > (\frac{cn}{\delta})^{\frac{1}{\epsilon}}] = \underset{x \leftarrow \mathcal{D}_n}{\Pr}[t_A^\epsilon(x, 1^n) > \frac{cn}{\delta}] < \delta$$

Последнее неравенство следует из неравенства Маркова.

Теперь покажем, что из определения Импальяццо следует определение Левина.

Пусть алгоритм $A(x, \delta, 1^n)$ решает распределенную задачу распознавания за полиномиальное в среднем время в определении Импальяццо. Построим алгоритм $B(x, 1^n)$ такой, что он решает эту же задачу за полиномиальное в среднем время в определении Левина. Алгоритм B запускает алгоритм на следующих входах: $A(x, \frac{1}{2}) \dots A(x, \frac{1}{2^k})$ пока не встретится первый ответ, отличный от $\perp$. Этот ответ B и выдает в качестве результата своей работы.

Докажем, что построенный алгоритм удовлетворяет определению Левина. Пусть время работы $(x, \delta, 1^n)$ ограничено $(\frac{n}{\delta})^c$ и пусть $\epsilon = \frac{1}{2c}$. Тогда:

$$\underset{x \leftarrow \mathcal{D}_n}{E}[t^{\epsilon}_B(x, 1^n)] \leq \sum_{k=1}^{\infty} (n2^k)^{\frac{1}{2}} \frac{1}{2^k} = \sum_{k=1}^{\infty} \frac{n^{\frac{1}{2}}}{2^{\frac{k}{2}}} = O(n^{\frac{1}{2}}) = O(n)$$

□

Введем определения нескольких сложностных классов, связанных со сложностью в среднем.

Определение 11.5. $AvgP$ — это класс распределенных задач распознавания $(L, \mathcal{D})$, которые решаются за полиномиальное в среднем время.

Мы также будем рассматривать немного похожий класс, класс задач, которые решаются за эвристическое полиномиальное время.

Определение 11.6. $HeurP$ — это класс распределенных задач распознавания $(L, \mathcal{D})$, для которых существует алгоритм $A(x, \delta, 1^n)$ такой, что:

1. Время его работы ограничено некоторым полиномом $p(\frac{|x|}{\delta})$
2. $\Pr\limits_{x \leftarrow \mathcal{D}_n}[A(x, \delta, 1^n) \neq L(x)] < \delta$

Легко видеть, что $AvgP \subseteq HeurP$ (достаточно ответ $\perp$ заменить на 0).

Определение 11.7. $Avg_{\delta(n)}P$ — это класс распределенных задач распознавания $(L, \mathcal{D})$, для которых существует алгоритм $A(x, 1^n)$ такой, что:

1. $A(x, 1^n)$ работает полиномиальное время.
2. $A(x, 1^n) \in \{L(x), \perp\}$
3. $\Pr\limits_{x \leftarrow \mathcal{D}_n}[A(x, 1^n) = \perp] < \delta(n)$

Определение 11.8. $Heur_{\delta(n)}P$ — это класс распределенных задач распознавания $(L, \mathcal{D})$, для которых существует алгоритм $A(x, 1^n)$ такой, что:

1. $A(x, 1^n)$ работает полиномиальное время.
2. $\Pr\limits_{x \leftarrow \mathcal{D}_n}[A(x, 1^n) \neq L(x)] < \delta(n)$

Легко видеть, что $AvgP \subseteq Avg_{\frac{1}{n^c}}P \subseteq Heur_{\frac{1}{n^c}}P$.

11.1 Пример распределения, для которого сложность в худшем случае и в среднем совпадают

В этом разделе мы покажем, что существует ансамбль распределений, по отношению к которому понятия сложности в среднем и в худшем случае совпадают. По этой причине при изучении сложности в среднем обычно не рассматривают все возможные распределения, а рассматривают только некоторые их классы.

Мы будем рассматривать пары (M, w), где M это машина Тьюринга, а w — строка. Напомним, что если длина записи M равна l, а длина записи w равна n, то длина записи (M, w) равна $l + n + 2\log l + 2\log n + O(1)$.

Для двоичной строки x обозначим через $K(x)$ длину самой короткой пары (M, w) такой, что машина M на входе w выдает x. Величина $K(x)$ называется префиксной колмогоровской сложностью строки x.

Универсальное вероятностное распределение K определяется так, чтобы вероятность строки x была равна $2^{-K(x)}$. Заметим, что $\sum_x 2^{-K(x)} \leq 1$, поскольку представление (M, w) префиксно свободное (на самом деле, $\sum_x 2^{-K(x)} \leq 1$, поэтому формально K не является вероятностным распределением, но мы можем его подправить, присвоив, например, пустой строке ϵ вероятность $1 - \sum_{x \neq 0} 2^{-K(x)}$). Наконец, пусть $\{K_n\}$ это будет ансамбль распределений, где K_n это распределение K только, ограниченное на строчки длины n.

Теорема 11.2. *Существует такой ансамбль распределений $\mathcal{D}$, что для любого разрешимого языка L если распределенная задача распознавания $(L, \mathcal{D})$лежит в $Heur_{\frac{1}{n^3}} P$, то $L \in P$.*

Доказательство. Мы будем использовать определенный выше ансамбль распределений $\{K_n\}$.

Пусть A это полниомиальный эвристический алгоритм, который подтверждает тот факт, что $(L, \{K_n\}) \in Heur_{\frac{1}{n^3}} P$. Покажем, что лишь для колечного числа строк $A(x, |x|) \neq L(x)$, из чего следует, что $L \in P$.

По определению:

$$K_n(x) = \frac{2^{-K(x)}}{\sum_{y \in \{0,1\}^n} 2^{-K(y)}}$$

и можно видеть, что $\sum_{y \in \{0,1\}^n} 2^{-K(y)} = \Omega(\frac{1}{n(\log n)^2})$, поскольку строка 0^n имеет колмогоровскую сложность не более, чем $\log n + 2\log\log n + O(1)$ и таким образом вносит вклад хотя бы $\Omega(\frac{1}{n(\log n)^2})$ в общую сумму. Из этого следует:

$$K_n(x) = O(n(\log n)^2 \cdot 2^{-K(x)}) = 2^{-K(x)+\log n+2\log\log n+O(1)}$$

Пусть теперь x это строка длины n такая, что $A(x, n) \neq L(x)$. Поскольку полная вероятность всех таких строк не более, чем $\frac{1}{n^3}$, то в частности мы имеем $K_x(x) \leq \frac{1}{n^3}$, и

$$K(x) = \log \frac{1}{K_n(x)} - \log n - 2\log\log n - O(1) \geq 2\log n - 2\log\log n - O(1).$$

Рассмотрим тогда первую лексикографически такую строчку $x \in \{0,1\}^n$ (если она есть), что $A(x,n) \neq L(x)$. Такая строка может быть вычислена алгоритмом, который получает n и вычисляет $A(x,n)$ и $L(x)$ для всех строк $x \in \{0,1\}^n$ и выводит лексикографически первый x такой, что $A(x,n) \neq L(x)$ (здесь мы используем предположение о разрешимости языка L). Существование такого алгоритма доказывает, что $K(x) \leq \log n + 2\log\log n + O(1)$, то есть для достаточно больших n это противоречит приведенному выше неравенству для $K(x)$.

Из этого мы делаем вывод о том, что существует только конечное число длин входов, на которых A и L отличаются, а значит и конечное число самих таких входов.

□