Пространственная сложность. Классы $\mathcal{P}$SPACE и $\mathcal{NP}$SPACE. Вероятностные классы сложности: $\mathcal{BPP}, \mathcal{RP}, \mathcal{ZPP}$. Семинары 5-6. 7 и 14 марта 2019 г. (конспект пополняется)

Подготовил: Горбунов Э.

Ключевые слова: классы $\mathcal{P}$SPACE и $\mathcal{NP}$SPACE, теорема Сэвича, классы $\mathcal{P}$SPACE-complete, $\mathcal{P}$SPACE-hard, $\mathcal{BPP}$, $\mathcal{ZPP}$, $\mathcal{RP}$, лемма Шварца-Зиппеля

Литература: [Кормен 1, §6], [Кормен 2, §5 и дополнение C], [Мусатов, Глава 5, §5.1 - 5.3]

Пространственная сложность

На ранних этапах развития компьютеров и вычислительной техники память была существенно ограничена и была чуть ли не самым дорогим элементом компьютера. Поэтому людям при написании программ приходилось существенно задумываться о том, сколько памяти их программа использует. Иными словами, возникла потребность в изучении алгоритмов не только с точки зрения времени их работы, но и с точки зрения расходуемой памяти. Чтобы изучать таким способам алгоритмы, нужна была математическая формализация, о которой мы немного поговорим в этом семинаре.

Во-первых, договоримся, что нас будет интересовать вопрос о памяти, дополнительной к той, что тратится на хранение входа.

Определение. Пусть $s(n)$ — неубывающая функция. Классом $\mathcal{D}\text{SPACE}(s(n))$ называется класс языков, которые можно распознать на детерминированной машине Тьюринга, которая на входе длины n использует $O(s(n))$ ячеек на рабочих лентах.

Определение. Пусть $s(n)$ — неубывающая функция. Классом $\mathcal{N}\text{SPACE}(s(n))$ называется класс языков, которые можно распознать на недетерминированной машине Тьюринга, которая на входе длины n использует $O(s(n))$ ячеек на рабочих лентах при любых исходах недетрминированного выбора.

Определение. $\mathcal{P}\text{SPACE} = \bigcup\limits_{k=0}^{\infty} \mathcal{D}\text{SPACE}(n^k)$, $\mathcal{NP}\text{SPACE} = \bigcup\limits_{k=0}^{\infty} \mathcal{N}\text{SPACE}(n^k)$.

Неформально говоря, $\mathcal{P}$SPACE — это класс языков, которые распознаются на полиномиальной памяти детерминированными машинами Тьюринга, а $\mathcal{NP}$SPACE — это класс языков, которые распознаются на полиномиальной памяти недетерминированными машинами Тьюринга. Если говорить ещё более неформально, то $\mathcal{P}$SPACE — это аналог класса $\mathcal{P}$, но относительно используемой памяти, а не времени, а $\mathcal{NP}$SPACE — это, соответственно, аналог класса $\mathcal{NP}$ в теории пространственной сложности.

Как введённые классы связаны с классами, введёнными ранее, и между собой? Во-первых, очевидно, что $\mathcal{P}\text{SPACE} \subseteq \mathcal{NP}\text{SPACE}$. Следующие 2 упражнения поясняют связь между $\mathcal{P}, \mathcal{NP}$ и $\mathcal{P}$SPACE.

Упражнение. Покажите, что $\mathcal{P} \subseteq \mathcal{P}\text{SPACE}$.

Решение. Пусть $L \in \mathcal{P}$. Это означает, что существует полиномиально вычислимая МТ M, что $x \in L \iff M(x) = 1$. Заметим, что если МТ работает на входе x время $T_M(x) \leq p(|x|)$, где p — некоторый полином, то за время $T_M(x)$ МТ M успеет побывать не больше, чем в $T_M(x)$ ячейках, а значит, не больше чем $p(|x|)$ ячеек будут задействованы. Следовательно, она использует полиномиальную память.

Упражнение. Покажите, что $\mathcal{NP} \subseteq \mathcal{P}\text{SPACE}$.

Решение. Вспомним определение $\mathcal{NP}$: $L \in \mathcal{NP}$ тогда и только тогда, когда существует полиномиально вычислимая МТ $R(x, y)$ и полином $q(x)$, что

$$x \in L \iff \exists y \in \Sigma^* : \ (|y| \leq q(|x|) \text{ и } R(x, y) = 1)\,.$$

Осталось понять, что полиномиальной памяти хватит, чтобы организовать процедуру полного перебора сертификатов. Пусть время работы $R(x, y)$ на входе (x, y) равно $T_R(x, y) \leq p(|x| + |y|)$. Рассмотрим машину

Тьюринга M, которая по входу x будет выделять у себя на рабочей ленте $q(|x|)$ ячеек под перебор сертификатов и $p(|x| + q(|x|))$ ячеек для работы (выделение этой памяти делается за полиномиальное время). В первых $q(|x|)$ ячейках M будет хранить текущий сертификат y (всего возможных сертификатов $1+|\Sigma|+|\Sigma|^2+\ldots+|\Sigma|^{q(|x|)} = \frac{|\Sigma|^{q(|x|)+1}-1}{|\Sigma|-1}$), а следующие $p(|x|+q(|x|))$ ячеек рабочей ленты будут использоваться для запуска машины Тьюринга $R(x,y)$, где y — текущее значение сертификата. Если на данном сертификате получаем ответ 1, то M останавливается и возвращает 1, в противном случае, она затирает всё, что напечатала в ячейках для имитации работы $R(x,y)$, и затем печатает в ячейках для перебора сертификата следующий сертификат. Таким образом, мы используем полиномиальную память и рано или поздно МТ выдаст ответ 1 (если хоть один нужный сертификат существует) либо переберёт все сертификаты и выдаст 0. Следовательно, мы можем на полиномиальной памяти определить, существует ли нужный сертификат, а значит, проверить, что слово принадлежит языку L. Отсюда следует, что $L \in \mathcal{P}$SPACE, а значит, в силу произвольности выбора L из $\mathcal{NP}$, получаем, что $\mathcal{NP} \subseteq \mathcal{P}$SPACE.

Упражнение. Покажите, что $\mathcal{P}$SPACE $\subseteq$ EXPTIME.

Решение. Чтобы это доказать, нужно заметить, что если машина Тьюринга на входе длины n использует не более $p(n)$ (дополнительной) памяти, то количество всевозможных конфигураций (напомню, что конфигурация — это символ, который читает головка на ленте в данный момент, и состояние головки) не превышает $a^{p(n)}p(n)nq$, где a — размер ленточного алфавита, q — мощность множества состояний машины Тьюринга: на рабочей ленте может быть написано $a^{p(n)}$ возможных слов, головка может быть в n позициях на входной ленте, в $p(n)$ позициях на выходной ленте и в q возможных состояниях. Если машина сделает $a^{p(n)}p(n)nq$ шагов и не остановится, то это означает, что в процессе работы некоторая конфигурация повториться, а значит, она будет повторяться бесконечное число раз, то есть МТ никогда не остановится. Следовательно, если МТ останавливается, то останавливается не более, чем за $a^{p(n)}p(n)nq$ шагов, а это как раз экспоненциальное время, так как $a^{p(n)}p(n)nq \le 2^{p(n)} \cdot 2^{p(n)nq} = 2^{\overbrace{p(n)(1+nq)}^{\text{полином}}}$. Следовательно, если $L \in \mathcal{P}$SPACE, то $L \in$ EXPTIME.

Оказывается, если ещё немного видоизменить доказательство, то можно показать, что $\mathcal{NP}$SPACE $\subseteq$ EXPTIME (детали локазательства можно прочитать в [Мусатов, Глава 5, §5.1 - 5.3]), т.е. верна

Теорема. $P \subseteq \mathcal{NP} \subseteq \mathcal{P}$SPACE $\subseteq \mathcal{NP}$SPACE $\subseteq$ EXPTIME.

Оказывается, верно даже $\mathcal{P}$SPACE $= \mathcal{NP}$SPACE (*следствие теоремы Сэвича*), что нельзя так легко доказать для соответствующих временных классов $\mathcal{P}$ и $\mathcal{NP}$ (напомню вопрос о равенстве классов $\mathcal{P}$ и $\mathcal{NP}$ до сих пор не решён). Более того, ничего неизвестно про равенство $\mathcal{NP}$ и $\mathcal{P}$SPACE.

Теорема. (Сэвич). Если $s(n) \ge \log n$, то $\mathcal{N}$SPACE$(s(n)) \subseteq \mathcal{D}$SPACE$(s(n)^2)$.

На доказательстве теоремы мы останавливаться не будем, т.к. при детальном рассмотрении треует отдельной лекции (доказательство можно прочитать, например, в [Мусатов, Глава 5, §5.1 - 5.3]). Отметим только, чтоб т.к. квадрат полинома — это тоже полином, из теоремы Сэвича вытекает

Следствие. $\mathcal{P}$SPACE $= \mathcal{NP}$SPACE.

Классы $\mathcal{P}$SPACE-hard и $\mathcal{P}$SPACE-complete. Примеры $\mathcal{P}$SPACE-полных языков.

Классы $\mathcal{P}$SPACE-hard и $\mathcal{P}$SPACE-complete определяются точно так же, как $\mathcal{NP}$-hard и $\mathcal{NP}$-complete.

Определение. $L \in \mathcal{P}$SPACE-hard $\iff \forall A \in \mathcal{P}$SPACE $\hookrightarrow A \le_P L$, где $\le_P$ — сводимость по Карпу, или просто полиномиальная сводимость.

Определение. $L \in \mathcal{P}$SPACE-complete $\iff \forall A \in \mathcal{P}$SPACE $\hookrightarrow A \le_P L$ и $L \in \mathcal{P}$SPACE.

Рассмотрим несколько примером $\mathcal{P}$SPACE-полных языков.

Определение. Язык SPACETMSAT есть множество $\{(M, x, 1^s) \mid M(x) = 1$ и $M(x)$ занимает не больше s ячеек памяти$\}$.

Теорема. SPACETMSAT$\in \mathcal{P}$SPACE-complete.

Доказательство. Первый этап состоит в докзательстве SPACETMSAT$\in \mathcal{P}$SPACE: запустим $M(x)$, ограничив зону работы величиной s и контролируя, что машина не зациклилась. Если машина остановилась и выдала 1, то выдаём 1. Если МТ выдала 0, попыталась выйти за пределы зоны или зациклилась, то выдаём 0. Построенный алгоритм использует $O(s)$ памяти, то есть полиномиален по памяти. Второй эта состоит в построении сведения произвольного языка $L \in \mathcal{P}$SPACE к SPACETMSAT. Если $L \in \mathcal{P}$SPACE, то существует МТ M, которая на входе x занимает память не больше $p(|x|)$. Рассмотрим сводящую функцию: $f(x) = (M, x, 1^{p(|x|)})$. По определению SPACETMSAT получаем, что $x \in L \iff f(x) \in$ SPACETMSAT, т.е. $L \leq_P$ SPACETMSAT, ибо сводящая функция полиномиально вычислима по времени.

Определение. Языком TQBF (“totally quantified boolean formulae”, или булевы формулы с кванторами, БФК) называется множество булевых формул φ, таких что для некоторого $x_1 \in \{0,1\}$ найдётся $x_2 \in \{0,1\}$, такое что для некоторого $x_3 \in \{0,1\}$ найдётся $\ldots$ (цепочка чередующихся кванторов по всем переменным) $\varphi(x_1, x_2, \ldots) = 1$.

Теорема. TQBF $\in \mathcal{P}$SPACE-complete.

Доказательство этой теоремы можно прочитать в [Мусатов, Глава 5, §5.1 - 5.3].

Вероятностные классы сложности

to be continued...