Лекция 5

Полиномиальная иерархия

(Конспект: А. Куликов)

5.1 Класс PSPACE.

Напомним, что класс **PSPACE** определяется следующим образом:

$$\mathbf{PSPACE} = \bigcup_{k \geq 0} \mathbf{DSpace}(O(n^k)).$$

Как и всякий «приличный» класс, **PSPACE** имеет полную задачу:

Теорема 5.1. `QBF` *(задача выполнимости формулы в КНФ с кванторами; здесь считаем формулу замкнутой, то есть кванторы стоят по всем переменным) —* **PSPACE***-полна.*

Доказательство. Рассмотрим произвольный язык, принадлежащий классу **PSPACE**, и машину Тьюринга с полиномиальной памятью, принимающую этот язык (такая машина есть как раз потому, что язык лежит в **PSPACE**). Ясно, что у такой машины есть не более, чем $2^{p(n)}$ различных конфигураций. Нас интересует, можем ли мы из начальной конфигурации попасть в принимающую за $2^{p(n)}$ шагов (ведь если можем, то такого количества шагов будет достаточно: если какая-то конфигурация повторится, машина уже никогда не остановится).

Для двух конфигураций c_1 и c_2 положим $\phi_i(c_1, c_2)$ = «существует путь из c_1 в c_2 длины не более, чем 2^i». Таким образом, для решения задачи нам необходимо найти значение функции $\phi_{p(n)}(c_0, c_Y)$, где c_0 — начальная, а c_Y — принимающая конфигурации. Запишем теперь рекуррентное соотношение:

$$\phi_i(c_1, c_2) = \exists d \forall x \forall y (((x = c_1 \wedge y = d) \vee (x = d \wedge y = c_2)) \Rightarrow \phi_{i-1}(x, y)).$$

(Значение функции $\phi_0(c_1, c_2)$ нужно записать аналогично тому, как это делается в теореме Кука-Левина.) Нетрудно видеть (вынося кванторы из формулы), что таким образом исходная задача может быть записана в виде формулы в ДНФ с кванторами. Свести полученную задачу (о формуле в ДНФ с кванторами) к задаче о формуле в КНФ с кванторами можно, воспользовавшись следующим очевидным соображением: $f \in \texttt{QBF} \Leftrightarrow \bar{f} \notin \texttt{QBF}$.

Тем самым мы показали, что QBF — **PSPACE**-трудна. Покажем теперь, что QBF $\in$ **PSPACE**.

Рассмотрим дерево расщепления (т.е. дерево, состоящее из всех наборов значений переменных, где двумя сыновьями вершины являются расширения набора $\vec{y} \leftarrow \vec{b}$ до наборов $\vec{y} \leftarrow \vec{b}, x \leftarrow 0$ и $\vec{y} \leftarrow \vec{b}, x \leftarrow 1$ для некоторой переменной x), в котором порядок расщепления соответствует порядку, в котором в начале формулы идут кванторы по этим переменным. В листьях запишем значение (бескванторной) формулы на соответствующем наборе значений. Пойдем по этому дереву снизу вверх и будем в каждой вершине, в которой произошло расщепление по переменной с квантором всеобщности, писать 1, если 1 написаны в обоих сыновьях этой вершины; для переменной существования будем писать 1, если 1 написана хотя бы в одном из сыновей. Ясно, что этот рекурсивный алгоритм, во-первых, решает поставленную задачу, а во-вторых, требует не более, чем полиномиальной памяти (хранить нам нужно лишь проверяемую ветку — иначе говоря, стек вызовов — и два последних значения). □

5.2 Полиномиальная иерархия.

Для начала напомним понятие машины Тьюринга с оракулом: для языка L машина Тьюринга (детерминированная или недетерминированная) M^L работает так: в любой момент она может может, выписав на специальную оракульную ленту строку x и перейдя в специальное оракульное состояние, получить (на этой же ленте) ответ на вопрос, принадлежит ли x языку L, за один шаг (а в остальном она работает как обычная машина). Под C^D, где C — некоторый класс языков, для которых имеются машины Тьюринга с естественным образом определяемыми аналогичными им оракульными машинами Тьюринга, а D — класс языков, имеющий полный язык L, мы будем понимать машины с оракулом L.

Замечание 5.1. Если в D нет полного языка, это обозначение не вполне определено. Если C не задается машинами Тьюринга (например, задается парами взаимодействующих машин Тьюринга), то надо пояснять, что понимается под оракульными вычислениями — и результат может получиться разный. (Пример: интерактивные вычисления **IP** дают то же множество языков, что и **PSPACE**, но известен оракул C, для которого $\mathbf{IP}^C \neq \mathbf{PSPACE}^C$ — именно потому, что **IP** задается вычислительным устройством, для которого оракульный аналог определяется иначе).

Определение 5.1. Уровни полиномиальной иерархии:

$$\Sigma^0\mathbf{P} = \Pi^0\mathbf{P} = \Delta^0\mathbf{P} = \mathbf{P},$$

$$\Sigma^{i+1}\mathbf{P} = \mathbf{NP}^{\Sigma^i\mathbf{P}},$$

$$\Pi^{i+1}\mathbf{P} = \mathbf{co\text{-}NP}^{\Sigma^i\mathbf{P}},$$

$$\Delta^{i+1}\mathbf{P} = \mathbf{P}^{\Sigma^i\mathbf{P}}.$$

Полиномиальная иерархия:

$$\mathbf{PH} = \bigcup_{i\geq 0} \Sigma^i\mathbf{P}.$$

Ниже будет показана корректность определения, то есть то, что для каждого i класс $\Sigma^i\mathbf{P}$ содержит полную задачу. Нетрудно видеть, что любой класс одного из уровней полиномиальной иерархии содержит все классы предыдущих уровней.

Теорема 5.2. $L \in \Sigma^{k+1}\mathbf{P} \Leftrightarrow \exists$ *полиномиально ограниченное отношение* $R \in \Pi^k\mathbf{P}$, *такое, что* $\forall x(x \in L \Leftrightarrow \exists y R(x, y))$.

Доказательство. Достаточность. Машиной, распознающей язык L, будет машина с оракулом R (понятно, что неважно, будет оракулом $\Sigma^k\mathbf{P}$, или же $\Pi^k\mathbf{P}$), недетерминированно выбирающая строку y и после этого проверяющая отношение $R(x, y)$ при помощи оракула.

Необходимость. Докажем утверждение по индукции. Рассмотрим недетерминированную машину M^O с оракулом $O \in \Sigma^k\mathbf{P}$, распознающую язык L. По предположению индукции из того, что $O \in \Sigma^k\mathbf{P}$, следует существование полиномиально ограниченного отношения $S \in \Pi^{k-1}\mathbf{P}$, такого, что $\forall q(q \in O \Leftrightarrow \exists z\, S(q, z))$.

Опишем теперь необходимое отношение R, то есть для каждой строки $x \in L$ построим соответствующую ей строку y. Рассмотрим принимающую (для входной строки $x \in L$) ветку дерева вычислений машины M^O и определим y следующим образом: пусть y кодирует эту принимающую ветку, причем для каждого обращения к оракулу (ведь мы рассматриваем вычисления на машине с оракулом!), на который последовал положительный ответ, y также содержит сертификат, подтверждающий принадлежность отношению (иными словами, для каждого $q \in O$, для которого рассматриваемая нами машина обращалась к оракулу, y содержит такое z, что $S(q, z)$).

Покажем, что описанное отношение принадлежит классу $\Pi^k\mathbf{P}$. Для проверки этого отношения необходимо проверить, во-первых, корректность всех шагов машины M^O (закодированных в строке y), что может быть сделано, очевидно, за полиномиальное время. Во-вторых, для всех пар (q, z) мы должны проверить отношение $S(q, z)$; при этом мы остаемся в переделах класса $\Pi^k\mathbf{P}$, так как $S \in \Pi^{k-1}\mathbf{P}$. И наконец, для каждого отрицательного обращения к оракулу, мы должны проверить, действительно ли проверяемая строка не принадлежит O; в силу того, что $O \in \Sigma^k\mathbf{P}$ и $\Sigma^k\mathbf{P} = \mathbf{co}\text{-}\Pi^k\mathbf{P}$, это может быть сделано в пределах класса $\Pi^k\mathbf{P}$. Заметим, что здесь мы можем внутри одного $\Pi^k\mathbf{P}$-вычисления воспользоваться другим $\Pi^k\mathbf{P}$-вычислением, поскольку принимать вход мы будем только если вспомогательное вычисление будет принимающим (т.е. мы две недетерминированных машины[1], у каждой из которых все листья должны соответствовать принимающим конфигурациям, комбинируем в одну, и у нее снова все листья — принимающие). □

Применяя последовательно доказанную выше теорему, можно получить следующий факт:

Следствие 5.1. $L \in \Sigma^k\mathbf{P} \Leftrightarrow \exists$ *полиномиально ограниченное* $(k+1)$*-арное отношение* $R \in P$, *такое, что* $\forall x (x \in L \Leftrightarrow \exists y_1 \forall y_2 \exists y_3 \ldots R(x, y_1, y_2, \ldots, y_k))$.

Следствием из этой теоремы является существование полной задачи в классе $\Sigma^k\mathbf{P}$.

Определение 5.2. $\mathtt{QBF}_k$ — это множество истинных формул вида

$$\exists X_1 \forall X_2 \exists X_3 ... X_k \, \phi,$$

где ϕ — формула в КНФ или ДНФ, а $\{X_i\}_{i=1}^k$ — разбиение множества переменных этой формулы (на непустые непересекающиеся подмножества).

Следствие 5.2. $\mathtt{QBF}_k$ — $\Sigma^k\mathbf{P}$*-полна.*

Доказательство. Надо всего лишь записать полиномиально вычислимый предикат R в виде булевой формулы. Это мы уже научились делать в теореме Кука-Левина. □

[1] Вспомним, что $\Pi^k\mathbf{P}$ задается недетерминированной машиной с оракулом.

Теперь постараемся понять, имеются ли **PH**-полные задачи. Ответа на этот вопрос мы не получим, но увидим, что влечет за собой существование такой задачи. Вообще, многие заключения теорем в теории сложности выглядят примерно так: «тогда $\mathbf{PH} = \Sigma^{j_0}\mathbf{P}$», что неформально произносится так: «полиномиальная иерархия коллапсирует до j_0-го уровня».

Теорема 5.3. *Если* $\Sigma^k\mathbf{P} = \Pi^k\mathbf{P}$, *то* $\mathbf{PH} = \Sigma^k\mathbf{P}$. *В частности, если* $\mathbf{P} = \mathbf{NP}$, *то* $\mathbf{PH} = \mathbf{P}$, *а если* $\mathbf{NP} = \mathbf{co\text{-}NP}$, *то* $\mathbf{PH} = \mathbf{NP}$.

Доказательство. Очевидно, достаточно показать следующую импликацию: $\Sigma^k\mathbf{P} = \Pi^k\mathbf{P} \Rightarrow \Sigma^{k+1}\mathbf{P} = \Pi^k\mathbf{P}$. С этой целью рассмотрим язык $L \in \Sigma^{k+1}\mathbf{P}$. По теореме 4.2 существует отношение R из $\Pi^k\mathbf{P}$ (а следовательно, и из $\Sigma^k\mathbf{P}$), такое, что $L = \{x : \exists y\, R(x,y)\}$. Значит, существует отношение $S \in \Pi^{k-1}\mathbf{P}$, такое, что $R(x,y) \Leftrightarrow \exists z\, S(x,y,z)$. То есть $x \in L \Leftrightarrow \exists y \exists z\, S(x,y,z)$, а это значит, что $L \in \Sigma^k\mathbf{P}$. □

Следствие 5.3. *Если существует* **PH**-*полная задача, то полиномиальная иерархия коллапсирует.*

Доказательство. Предположим, такой язык L существует. Должно быть некоторое конкретное k, для которого $L \in \Sigma^k\mathbf{P}$. По условию следствия, любой язык $L' \in \mathbf{PH}$ сводится к L, то есть $L' \in \Sigma^k\mathbf{P}$ (ведь все уровни полиномиальной иерархии замкнуты относительно сведения). □