Домашнее задание 4, Павливского Сергея, 873.

0. Прочитайте в конспекте про полиномиальную иерархию (до пространственной сложности, про $\mathcal{PSPACE}$ пока можно не читать).

1. (i) Докажите, что в Σ_2 лежит язык булевых формул от двух наборов переменных $\varphi(x_1, \ldots, x_n, y_1 \ldots y_n) = \varphi(\vec{x}, \vec{y})$ таких, что при некоторых значениях $\vec{x}$ они справедливы вне зависимости от значений $y_1, \ldots, y_n$.

Решение:

Собственно, в условии и записано определение Σ_2 - существует МТ, для которой при заданной булевой формуле существует сертификат $(x_1, ..., x_n)$, для которого для любого сертификата $(y_1, ..., y_n)$ булева формула разрешается данной МТ.

(ii) Придумайте какую-нибудь свою задачу из класса Σ_3 (или Π_3, на ваш вкус).

Решение:

Ну, например, булева формула от трех наборов переменных $\varphi(x_1, ..., x_n, y_1$ $\varphi(\vec{x}, \vec{y}, \vec{z})$ таких, что для $\vec{x}$ для любого $\vec{y}$ существует $\vec{z}$, такой что φ полиномиально вычислима. Пример так же в точности соответствует определению требуемого языка Σ_3.

(iii) Докажите, что $\Sigma_k \subset \Sigma_{k+1} \cap \Pi_{k+1}$.

Решение:

Σ_k очевидно принадлежит Σ_{k+1} (R_k может просто игнорировать y_{k+1} верификатор). По аналогичным причинам Σ_k принадлежит $\Pi k + 1$. Значит $\Sigma_k \subset \Sigma_{k+1} \cap \Pi_{k+1}$.

Что и требовалось

2 (Доп). Покажите, как свести следующую задачу к вычислению некоторого перманента: найти количество перестановок n элементов, в которых части элементов (с номерами $i_1, i_2, \ldots i_k$) запрещено занимать позиции $j_1, \ldots j_k$ соответственно.

Решение:

Возьмем матрицу A $n x n$ такую, что если i-й элемент может стоять на j-й позиции, то $A[i][j] = 1$, иначе $A[i][j] = 0$. Тогда из определения разложения перманента по строке, перманентом как раз и будет являться число таких комбинаций, что ни один из элементов не стоит на запрещенном месте (иначе соответствующий ему элемент в перманенте равен 0, и всеь перманент равен 0).

$\mathrm{Per}(A) = \sum_{\alpha_1,\alpha_2,\ldots,\alpha_n} a_{1\alpha_1} a_{2\alpha_2} \ldots a_{n\alpha_n}$

3. а) Верно ли что язык 5-ДНФ-Л является полиномиально полным в $\mathbf{co}\text{–}\mathcal{NP}$?

Язык 5-ДНФ-Л состоит из всех формул в дизъюнктивной нормальной форме, принимающих истинное значение при каких-то значениях переменных, в каждый конъюнкт которых входит не более пяти переменных.

Решение:

Нет, не верно. 5-ДНФ-Л $\in$ P (аналогично пункту в)). Тогда если бы он был co-NP-c, то все co-NP языки бы разрешались за полином, т. е. P было бы равно co-NP, а значит и равно NP, что неверно в силу официальных гипотез.

б) Верно ли что язык 5-КНФ-Л является полиномиально полным в $\mathcal{NP}$?

Язык 5-КНФ-Л состоит из всех формул в конъюнктивной нормальной форме, принимающих ложное значение при каких-то значениях переменных, в каждый дизъюнкт которых входит не более пяти переменных.

Решение:

Как показано в пункте в) проверка КНФ (а вместе с ней и сводящейся в к ней 5-КНФ-Л) на тавтологичность $\in$ P. Тогда из 5-КНФ-Л $\in$ NP$-$c следовало бы P $=$ NP, что противоречит официальной гипотезе.

Можно использовать гипотезы $\mathcal{P}\mathrm{not} = \mathcal{NP}$ *и* $\mathcal{NP}\mathrm{not} = \boldsymbol{co}\text{–}\mathcal{NP}$.

в) Расставьте и обоснуйте $\mathcal{P}$, $\mathcal{NP} - \mathrm{complete}$, $\mathrm{co} - \mathcal{NP} - \mathrm{complete}$:

	Выполнимость	Тавтологичность
КНФ	NP-c	P
ДНФ	P	co-NP-c

Под выполнимостью понимается задача проверки наличия набора значений переменных, на котором формула равна 1. Под тавтологичностью понимается задача проверки свойства формулы принимать значение 1 на всех наборах.

CNF-SAT $\in$ NP-c (обсуждалось на семинарах) - выполнимость (ну или просто сказать, что 3CNF к ней сводится мощнейшей формулой, которая не меняет ничего в CNF)

DNF-выполнимость $\in$ P, так как можно просто за квадрат перебрать все элементы каждого конъюнкта, проверив их на непротиворечивость, и за линию повторить это доля каждого конъюнкта (итого за куб).

Тавтологичность CNF сводится за полином к задаче выполнимости DNF взятием отрицания от CNF: отрицание CNF, очевидно, выполнимо тогда и только тогда, когда исходная CNF не тавтологична.

Аналогично, сведем к задаче тавтологичности DNF задачу выполнимости КНФ: отрицание NP-c - это co-NP-c, т.е. мы можем свести co-NP-c к задаче тавтологичности DNF, т. е. эта задача co-NP-hard. Если теперь также заметить, что эта задача $\in$ co-NP (сертификат- набор значений, на которых формула обращается в 0), то отсюда следует, что задача тавтологичности DNF co-NP-c.

4. Найдите Θ-асимптотику суммы $\sum\limits_{k=1}^{n} \sqrt{k}$, оценив её с помощью интеграла $\displaystyle\int_1^n \sqrt{x}dx$ сверху и снизу. Выведите аналогичную формулу для асимптотики $\sum\limits_{k=1}^{n} k^{\alpha}$ для $\alpha > 0$.

Решение:

Из графика функции $y = \sqrt{x}$ видно , что из ее монотонного роста и всюду одинаковой выпуклости $\int_{k-1}^{k} \sqrt{x}dx < \sqrt{k} < \int_{k}^{k+1} \sqrt{x}dx$ (1) (действительно, первый интеграл ограничивается сверху площадью прямоугольника с высотой равной верхней границы интеграла и основанием равным длине интегрируемого промежутка, в то время как сам прямоугольник ограничивается сверху площадью криволинейной трапеции с таким же основанием, такой же левой верхней границей, что и прямоугольника, но у которой верхняя граница возрастает). Тогда заметим, что $\int_1^n \sqrt{x}dx = \sum_{k=1}^{n-1} \int_k^{k+1} \sqrt{x}dx$. Тогда суммируя неравенство (1) по k от 1 до n имеем:

$\frac{2}{3}n^{\frac{3}{2}} = \int_0^n \sqrt{x}dx = \sum_{k=1}^n \sqrt{k} = \int_1^{n+1} \sqrt{x}dx = \frac{2}{3}(n+1)^{\frac{3}{2}} - \frac{2}{3} < \frac{2}{3}(n+1)^{\frac{3}{2}} < \frac{2}{3}(2n)^{\frac{3}{2}} < 2^{\frac{5}{2}}n^{\frac{3}{2}}$

$\sum_{k=1}^n \sqrt{k} = \theta(n^{\frac{3}{2}}$

По аналогии, $\forall\alpha > 0 \Rightarrow \int_{k-1}^k x^\alpha dx < k^\alpha < \int_k^{k+1} x^\alpha dx$. Тогда действуя так же получаем, что $\sum_{k=1}^n k^\alpha = \theta(n^{\alpha+1})$.

Что и требовалось

5. Останется ли $3 - SAT$ полной, если ограничиться формулами, в которых каждая переменная входит не более 3 раз, а каждый литерал — не более 2 раз?

а) Под $3 - SAT$ понимается НЕ-БОЛЕЕ-$3 - SAT$.

Решение:

Извернемся через хвост:

Нам нужно, чтобы каждая переменная входила не более 3 раз- тогда заменим часть вхождений каждой переменной в исходную формулу на новую переменную, и дополним формулу конструкцией , которая будет накладывать связь на новые переменные . Необходима выполнимость формулы только если каждая переменная по значению равна замененному литералу - сделаем кольцевую конструкцию $(x_i \vee \overline{a_{i0}}) \wedge (a_{i0} \vee \overline{a_{i1}}) \wedge ... \wedge (a_{ik} \vee \overline{x_i})$. Видно, что она выполняется лишь когда все переменные в нее входящие равны. Она содержит 2 вхождения исходной переменной, по 2 вхождения новой переменной, тогда в исходной формуле можно заменять все x_i кроме одного, тогда всех переменных после дополнения формулы вышеописанной конструкцией будет 3 переменных, и, поскольку в конструкцию для каждой переменной входит по 2 разных литерала, то по принципу Дирихле в объединенной формуле будет не более 2-х вхождений одного литерала и не более 3-х вхождений одной переменной. Т. е. сводимость:

- для каждой переменной все ее вхождения кроме одного заменить на соответствующую переменную

- построить вышеописанную конструкцию

- сделать конъюнкцию формулы с замененными переменными с конструкцией

Если исходная формула выполняется, то в новой возьмем значения заменных переменных такие же, как и у литерала, от которого они были образованы. Тогда новая формула также выполняется.

Если новая формула выполняется, то все переменные равны своим заменам, значит исходная формула выполняется на поднаборе $\{x_i\}_{i=1}^n$.

Что и требовалось

б) (**Бонусная задача**) Покажите, что если имеется в виду РОВНО-$3-SAT$, то не бывает невыполнимых формул указанного вида.

6. Постройте сводимость по Карпу языка (G, k) графов, в которых есть k-клика к языку графов, в которых есть клика хотя бы на половине вершин.

Решение:

Пусть в графе n вершин. Добавим еще $n-2k$ вершин, и соединим их со всеми остальными.

Докажем корректность сводимости.

$\Rightarrow$ если в исходном графе есть k-клика , то, т. к. все добавленные вершины будут соединены с вершинами этой клики, то в новом графе будет клика на новых вершинах и k старых из $n-2k+k=n-k$ вершин , т. е. хотя бы половина вершин нового графа.

$\Leftarrow$ если в новом графе есть клика из хотя бы половины вершин, т. е. $n-k$ вершин, то хотя бы k вершин из них являются вершинами исходного графа (если их $< k$, то, т. к. новых $n-2k$, то всего вершин в клике $< n-2k+k = n-k$?!), а тогда по определению клики они все соединены между собой, т. е. образуют в исходном графе k-клику.

Что и требовалось

7. Покажите, что если всякий $\mathcal{NP}$-трудный язык является $\mathcal{PSPACE}$-трудным, то $\mathcal{NP} = \mathcal{PSPACE}$.

Решение:

Так как любой NP-c язык также NP-hard, то найдется NP язык, который PSPACE-hard (по определению NP-c). Тогда от любого PSPACE языка мы можем вместе с ним подать на вход сертификат для соответсвующего NP языка, вычислить полиномиальную сводящую функцию, затем с помощью сертификата проверить принадлежность NP языку за полином, которая будет равносильная принадлежности исходного языка PSPACE. Тогда любой PSPACE язык NP по определению.

Любой NP является PSPACE (иерархия была на лекциях; ну или можно сказать, что PSPACE не задает условия на время работы, поэтому можно взять полиномиальный от длины слова промежуток памяти и перебрать на нем перебрать все варианты, ибо МТ для NP использует полиномиальное число ячеек памяти , т. к. на задействование одной ячейки нужен хотя бы один такт МТ, и в случае неполиномиального числа ячеек памяти и время работы было бы неполиномиальным).

Из включения в обе стороны следует равенство языков.