ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
МОСКОВСКИЙ ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Алгоритмы и модели вычислений.
Домашняя работа 3.

Георгий Никишин
Группа Б05-909

Задача 1.

Докажите, что если $A \leq_p B$, то $\overline{A} \leq_p \overline{B}$.

Решение.

Вспомним определение полиномиальной сводимости. $A \leq_p B$, если $\exists$ полиномиально вычислимая функция $f(x)$: $x \in A \Leftrightarrow f(x) \in B$. Тогда для сведения $\overline{A} \leq_p \overline{B}$ возьмем в качестве функции ту же $f(x)$, что и в $A \leq_p B$, тогда $x \notin A \Leftrightarrow f(x) \notin B \Rightarrow x \in \overline{A} \Leftrightarrow f(x) \in \overline{B}$, что и означает, что $A \leq_p B$.

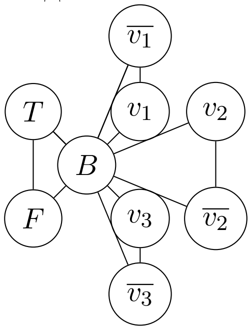
Задача 2.

Покажите, что если язык 3COL $\in$ P, то за полиномиальное время можно не только определить, что граф допускает раскраску в три цвета, но и найти какую-то 3-раскраску (если она существует).

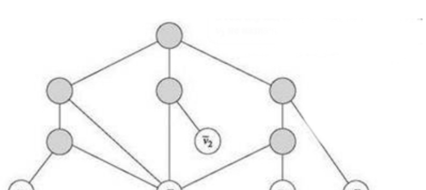
Решение.

Сначала докажем, что 3COL $\in$ NPC.

1) Докажем, что 3COL $\in$ NPH. Сведем разобранную на саминаре задачу 3CNF из класса NPC к нашей. Для этого постороим граф по исходной 3-кнф. Определим вершины графа v_i и $\overline{v_i}$, соответсвующие переменным x_i и $\overline{x_i}$. Далее определим три базовых узла True, False и Base. Для начала мы соединим каждую пару узлов $v_i, \overline{v_i}$ ребром и соединим оба этих узла с Base (в результате чего образуется треугольник из $\{v_i, \overline{v_i}, Base\}$, далее соединим True, False и Base. Граф, имеющийся у нас на данный момент выглядит так:



В нашей раскраске True, False и Base должны быть присовены три разных цвета в произвольном порядке, кроме того, v_i и $\overline{v_i}$ должны быть присвоены различные цвета, каждый из который должене отличаться от цвета Base. В частности, это означает, что для всех i одно из значений v_i или $\overline{v_i}$ получает цвет True, а другому достается цвет False. В оставшейся части этого построения будем считать, что переменной x_i значение 1 в заданном экземпляре 3-SAT присваивается в том, и только в том случае, если вершине v_i присвоен цвет True. Тогда мы получили граф G, в котором любая 3-раскраска неявно определяет логическое присваивание для переменных в экземпляре 3-SAT. Теперь необходимо нарастить G так, чтобы только выполняющие присваивания могли быть расширены до 3-раскрасок полного графа. Рассмотрим например условие вида $x_1 \vee \overline{x_2} \vee x_3$, значит, что хотя бы одна из $x_1, \overline{x_2}$,x_3 - истинна, то есть хотя бы одной из соответствующим им вершин назначен цвет True. Нам нужен маленький подграф, который можно присоединить к G, чтобы любая 3-раскраска, расширяющаяся на этот подграф, обладала свойством назначения цвета True по крайней мере одной из вершин $v_1, \overline{v_2}, v_3$



Этот подграф из шести узлов "присоединяется" к основному графу G в пяти существующих узлах: True, False и узлах, соответствующих трем литералам в условии, которое мы пытаемся представить. Тепрь предположим, рассматриваемым узлам в нашей раскраске одновременно назначен цвет False, тогда двум темным нижним вершинам назначен Base, трем темным узлам над ними - F, B,T-

слева-направо.Тогда верхний узел не может быть раскрашен $\Rightarrow$ хотя бы одна из вершин-True. Легко убедиться, что если хотя бы одна из вершин-T , то все хорошо и подграф- 3-раскрашиваем.Далее остается лишь завершить построение: мы начинаем с графа G, определенного выше, и для каждого условия в экземпляре 3-SAT присоединяем подграф из шести узлов, изображенный выше. Докажем, что построенный подграф 3-раскрашиваем $\Leftrightarrow$ соотв 3-кнф выполнима.
$\Leftarrow$ Если 3-кнф выполнима, то $\forall i$ назначаем v_i T, если $x_i = 1$, v_i F, если $x_i = 0$, в соотв наборе переменнных, обращающий 3-кнф в истину. Тогда по построению $\overline{v_i}$ назначается единственный возможный цвет, после чего раскраска расширяется на каждый подграф.
$\Rightarrow$ Теперь пусть соотв раскраска графа, докажем, что есть набор переменных, на которых кнф выполняется. Предположим, что такого набора нет, тогда все дизъюнкты обнуляются $\Rightarrow$ всем вершинам этого дизъюнкта присвоен False, но тогда граф не 3-раскрашиваем, это доказывается аналогично пункту в построении. Тогда получили противоречие. $\Rightarrow$ 3-COL $\in$ NPH.
Докажем, что 3-COL $\in$ NP. Для этого в качестве сертификата будем использовать саму раскраску графа, а верификатором будет служить алгоритм, проверяющий, что смежные вергины покрашены в разные цвета, очевидно, что подсказка есть полином от числа вершин графа, кроме того, алгоритм также полиномиален $\Rightarrow$ 3COL $\in$NP. $\Rightarrow$ 3COL $\in$ NPC.
Тк язык **3COL** $\in NPC$, то если допустить, что **3COL** $\in P$, то получим, что $P = NP$, тк к **3COL** полиномиально сводится $\forall$ задача из NP, а **3COL** $\in P$, тогда $\forall$задача из NP решается за полином. Теперь докажем, что поиск 3-раскраски графа является NP задачей. В качестве сертификата будем использовать саму раскраску, а в качестве верификатора-алгоритм, который будет проверять, нет ли конфликтов в раскрашенном таким образом графе. Тк подсказка имеет длину $O(V)$, а алгоритм проверки работает за полином от числа вершин, то задача будет лежать в NP, тогда если предположить, что **3COL** $\in P$, то и задача о нахождении раскраски $\in P$.

Задача 3.a

Докажите NP-полноту языка **CLIQUE{G,k}**,

Решение.

Сначала докажем, что аналогичная задача о независимом мноджестве размера k является NP-трудной, для этого покажем, что $3SAT \leq_p INDSET$. Построим граф по КНФ следующим образом: каждлому вхождению литерала в КНФ сопоставим вершину в графе. Таким образом, если в исходной КНФ k дизъюнктов, то в графе будет $3k$ вершин. Соединим ребрами те вершины, которые соответсвуют одному дизъюнкту в нашей КНФ. Кроме того, соединим все вершины, соответствующие противоположным литералам (x_i и $\overline{x_i}$)
Докажем, что данный граф эквивалентен 3-КНФ.
$\Leftarrow$ Пусть 3-КНФ выполнима, тогда в каждом дизъюнкте хотя бы один литерал истенен, тогда соответсвующие этим литералам образуют в построенном графе независимое множество, тк мы проводили ребра только между литералами из одного дизъюнкта и противоположными. Тк x_i и $\overline{x_i}$ не могут быть истинны одновременно, тогда есть независимое множество размера k.
$\Rightarrow$ Пусть есть независимое множество размера k, тогда по построению эти вершины должны соответствовать различным дизъюнктам, при этом тк среди них нет одновременно литералов x и $\overline{x}$, то можно взять значения переменных, при которых все используемые литералы будут истинны. Тогда на этих значениях наша 3-КНФ истинна.
Докажем, что INDSET $\in$ P. В качестве сертификата возьмем матрицу смежности графа и номера вершин, образующих клику , тогда верифицируем наш сертификат алгоритмом, который будет проверять, соединены ли помеченные вершнины ребром. Тогда посылка есть полином от длины входа, а алгоритм работает за полином, тк просто проверяет число в матрице смежности по указанному адресу$\Rightarrow$ Тогда, тк если граф имеет независимое множество размера k, то его дополнение имеет клику того же размера, задача отклике эквивалента задаче о независимом множестве $\Rightarrow$ задача о клике - NPC.
Ремарка, нужно сказать, что если мы умеем решать задачу о клике размера k, то мы можем решать ее и для любого другого числа большего k, кроме того, для пренадлежности графа языку, нам до-

статочно найти клику размера k, даже если есть клика большего размера.

Задача 3.б

Доказать NP-полноту языка $\{G| \;\; w(G) \geq \frac{9}{10}|V(G)|\}$

Решение.

Полностью аналогично решанию пункта а, за исключением того, что для каждого графа будем брать свое $k = \frac{9}{10}|V(G)|$

Задача 4

Назовём подмножество вершин C графа G его вершинным покрытием, если хотя бы один конец каждого ребра графа G принадлежит C.

Докажите NP-полноту языка VERTEXCOVER = (G, k) | в графе G есть вершинное покрытие на (не более чем) k вершинах.

Решение.

Воспользуемся тем, доказанным в задаче 3а утверждением. Мы знаем, что задача о независимом множестве $INDSET\{G,k\}$ является NP-полной. Тогда сведем задачу о независимом множестве к нашей. Пусть в графе выбрано независимое множество вершин $\overline{C}$, тогда, согласно определению вершинного покрытия, $\forall$ ребра графа хотя бы одна вершина не лежит в $\overline{C}$. Тогда посмотрим на дополнение к $\overline{C}$, понятно, что оно является вершинным покрытием графа. Теперь покажем обратное. Пусть в графе выбрано вершинное покрытие C, тогда по определению любому ребру графа инцидентна хотя бы одна вершина из C.Тогда аналогично расммотрим дополнение C, в нем нет ребер, соединяющих две вершины лежащие в $\overline{C}$. $\Rightarrow \overline{C}-$ независимое множество.

Тогда в графе существует независимое множество на k вершинах $\Leftrightarrow$ когда существует вершинное покрытие на $|V|-k$ вершинах. Тогда очевидно, что задачи сводятся друг к другу за полиномиальное время. Теперь для доказательства NP-полноты нам осталось доказать, что задача о вершинном покрытии лежит в NP, для решения в качестве сертификата возьмем набор из k вершин, образующих вершинное покрытие, тогда сможем верифицировать этот набор алгоритмом, который будет для каждого ребра проверять, есть ли в данном наборе хотя бы одна вершина , инцидентная ему. Для данной проверки понадобится $O(n^3)$ тк количество ребер есть $O(n^2)$, а для каждого ребра нужно проверить k вершин, здесь $n = |V|$.

Тогда мы получили, что задача о вершинном покрытии есть NPC.

Задача 5

Назовём подмножество вершин D графа G его доминирующим множеством, если каждая вершина, не принадлежащая D, смежна хотя бы с одной вершиной из D.

Докажите NP-полноту языка DOMSET = (G,k) | в графе G есть доминирующее множества размера не более k.

Решение.

Воспользуемся только что доказанным утверждением о том, что задача о независимом множестве есть NPC и сведем ее к нашей задаче, тем самым, доказав, что задача о доминирующем подмножестве есть NPC. Сделаем это следующим образом:

Рассмотрим исходный граф G и для каждого ребра $\{u;v\} \in E$ добавим в V новую вершину $\{uv\}$, после чего соединим ее с u и v. ПонятноE что такой алгоритм работает за полиномильное время от числа вершин, если быть точным за $O(|V|+|E|) = O(|V|^2)$, тк мы перебираем все ребра количество которых не превосходит квадрата числа вершин.

Теперь докажем, что данный алгоритм действительно сводит задачу о вершинном покрытии к задаче о доминантном множестве. Пусть исходный граф имеет вершинное покрытие C размера k, тогда каждое ребро в исходном графе имеет хотя бы одну вершину принадлежащую C $\Rightarrow \forall\{u;v\} \in E \rightarrow (u \in C \vee v \in C)$, тогда если $u \in C$, то v также покрывается некоторыми вершинами из C, аналогично если $v \in C$, тогда добавленные вершины в модифицированном графе также покрываются C, тк они смежны как с u , так и с v. Тогда множество вершин, образующий вершинное покрытие размера k в исходном графе также образуют и доминирующее множество в модифицированном графе. $\Rightarrow$ если

исходный граф имеет некоторое покрытие вершин, то модифицированный граф имеет доминирующее множество того же размера.
Докажем обратное, пусть модифицированный граф имеет доминирующее подмножество D размера k, тогда есть два варианта:
1) вершина, лежащая в D, принадлежит множеству вершин исходного графа
2) вершина является добавленной в процессе работы алгоритма
В первом случае все ок, во втором случае заменим нашу вершину uv на одну из вершин, смежных с ней, то есть на u или на v, заметим, что от этого вершинное покрытие никак не изменится, действительно, наша вершина uv смежна только с u и v, тогда при замене мы сможем охватывать все вершины, которые были до замены. Тогда заметим, что полученное множество вершин является как доминантным для модифицированного графа, так и покрывающим для исходного, тк для каждого ребра хотя бы одна из вершин обязана лежать в нем после замены. Тогда если у модифицированного графа есть доминирующее множество размера k, то у исходного графа есть покрывающее множество того же размера. Таким образом, данные задачи эквивалентны $DOMSET \in NPH$
Осталось доказать что данная задача является NP.
Для этого в качестве сертификата будем использовать множество вершин, образующих доминантное множество, а в качестве сертификата-алгоритм, проверяющий все вершины графа на смежность с хотя бы одной вершиной из этого множества, алгоритм полиномиален, тк для каждой вершины нам нужно перебрать не более k вершин $\Rightarrow$ он работает за $O(|V|^2)$

Задача 6

Язык HAMPATH состоит из описаний графов, имеющих гамильтонов путь. Язык HAMCYCLE состоит из описаний графов, имеющих гамильтонов цикл (проходя- щий через все вершины, причем все вершины в этом цикле, кроме первой и послед- ней, попарно различны). Постройте явные полиномиальные сводимости HAMCYCLE к HAMPATH и HAMPATH к HAMCYCLE.

Решение.

Для начала сведем задачу о пути к задаче о цикле пусть изначально в графе G есть гамильтонов путь, тогда модифицируем граф G. Добавим одну вершину и соединим ее с остальными, тогда в исходном графе есть гамильтонов путь $\Leftrightarrow$ в модифицированном графе есть гамильтонов путь, тк новая вершина будет соединена как с началом пути, так и с его концом.
Теперь сведем задачу о цикле к задаче о пути. Сделаем это следующим образом: "разорвем"наш цикл, возьмем ребро $\{u; v\}$ добавим вершины u' и v': u' соединена только с u, а v' только с v , тогда в полученном графе есть гамильтонов путь $\Leftrightarrow$ в исходном цикле есть гамильтонов цикл.
Обе задачи сводятся друг к другу за полиномильаное время, тк мы просто изменяем матрицу смежности, добавляя одну или две строчки в матрицу смежности.