1 Доказать, что если $L_2 \in \mathbf{NP}$, $L_1 \leq_p L_2 \ \Rightarrow \ L_1 \in \mathbf{NP}$

Т.к. у нас р-сводимость, за полином мы сводим язык L_1 к языку L_2. А поскольку $L_2 \in \mathbf{NP}$, то $\exists$ сертификат y. Если он подходит, то слово принадлежит языку. В противном случае - не принадлежит. И поскольку мы пользовались сводимостью, принадлежность $f(x) \in L_2 \Leftrightarrow x \in L_1$.
Можно считать, что всё это выполняет проверяющая машина Тьюринга, которая совершает $O(poly(x))$ операций (т.к. там только полиномиальная сводимость и полиномиальная проверка сертификата у $\mathbf{NP}$ -языка). Следовательно $L_1 \in \mathbf{NP}$

2 Доказать замкнутость NP относительно итерации Клини

Дан язык $L \in \mathbf{NP}$. $\forall x \in L^*$ будем перебирать всевозможные разбиения $T = \{x_i\}_{i=1}^n$ (которых $O(|x|^2)$, с.м. задачу 2.2 из дз3. Хотя можно перебирать и как-нибудь по-другому). И для удачных разбиений, т.е. состоящих только из слов языка L будем строить сертификат по следующему правилу:
Для подслова $x_i : x_i \in L$ допишем в сертификат у слова x строку вида $\#y_i@|x_i|\#$ где $\#$ и $@$ разделители, которых не было в алфавите языка. Т.е. итоговый сертификат будет содержать в себе последовательно сертификат подслова (оно ведь лежит в языке, который по условию $\in \mathbf{NP}$). Значит, если удачное разбиение существует, то мы можем разбить его на подслова $\in \mathbf{NP}$ и проверить каждый сертификат поотдельности. Например:
$L = \mathbb{N}/Primes$ - язык составных чисел. x=406327 - само по себе простое, но $x \in L^*$ потому что существуют удачные разбиения, например $x = \underbrace{40}_{y=2}\underbrace{63}_{y=3}\underbrace{27}_{y=3} = \underbrace{406}_{y=2}\underbrace{327}_{y=3}$.
Тогда сертификат для такого числа будет выглядеть так: $y = \underbrace{\#2@2\#}_{40}\underbrace{\#3@2\#}_{63}\underbrace{\#3@2\#}_{27}$ или при втором разбиении: $\underbrace{\#2@3\#}_{406}\underbrace{\#3@3\#}_{327}$.
Далее мы можем спокойно проверить каждое подслово (их мы можем выделить, зная длину, которая указана после @) за полиномиальное время по его сертификату, и просто проверить конкретное разбиение. Т.к. подслов не больше, чем длина слова x, то и сложность будет $O(poly(\log x))$
Если же слово не принадлежит итерации языка, то мы не сможем удачно его разбить на подслова, и сертификата не будет. Например, слово $x = 2311$ никак не разбивается на подслова из языка составных чисел, значит не принадлежит L^*.
Таким образом мы доказали замкнутость **NP** относительно итерации Клини.

3 Задача про coNP

3.1 Сформулируйте определение coNP (без ссылки на NP)

Если язык является **coNP** , то существует машина Тьюринга M и полином $poly$:
$L = \{x \in \Sigma^* : \forall y : |y| \leq poly(|x|) \ \Rightarrow \ M(x,y) = 0\}$
Или аналогично семинарскому: $L(x) = \exists[|y| \leq poly(|x|) \cap M(x,y) = 0]$, где $M(\cdot,\cdot) \in \mathbf{P}$ Т.е. аналогично определению **NP** ,

3.2 Доказать: $\mathbf{P} \subset \mathbf{coNP}$

$\forall L \in \mathbf{P}$ $\exists$ МТ M : она распознаёт язык за полиномиальное время. Т.к. это распознавание, то она же может распознать и отрицание языка (т.е. для $M(x) = 1$ верно $x \in L$ и $x \notin \overline{L}$ и наоборот). Т.е. $\forall L \in \mathbf{P} \Rightarrow \overline{L} \in \mathbf{P}$.
Далее, поскольку $\mathbf{P} \subseteq \mathbf{NP}$ значит $\overline{L} \in \mathbf{NP}$.
По определению $\mathbf{coNP}$ получим: $\overline{L} \in \mathbf{NP} \Leftrightarrow L \in \mathbf{coNP}$.
Получается, что $\forall L \in \mathbf{P}$ мы получили $L \in \mathbf{coNP}$. И раз все элементы одного множества ($\mathbf{P}$) принадлежат другому ($\mathbf{coNP}$), а это значит включение $\mathbf{P} \subset \mathbf{coNP}$

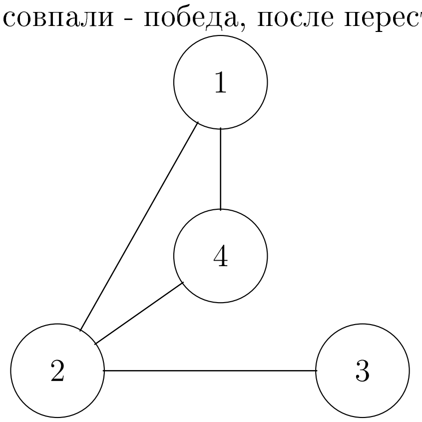
4 Доказать, что следующие задачи лежат в NP

4.1 Максимальная клика в графе имеет размер не меньше k

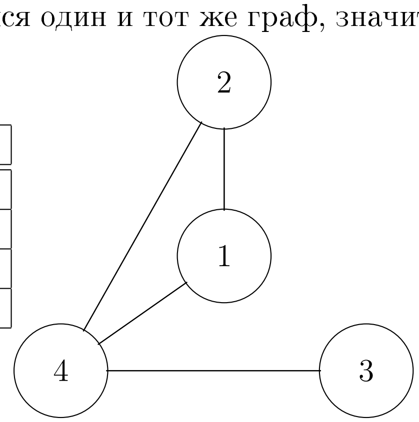
Пусть граф из $n = |V|$ вершин задан матрицей смежности a, т.е. длина входа n^2. Для проверки на принадлежность слова языку нам достаточно сертификата, в котором содержатся k вершин $s_1 \dots s_k$, составляющих клику в графе. Т.е. максимальный полный подграф.
Проверка на полноту этого подграфа осуществляется так: $\forall i, j : i \neq j$ просто проверяем существование хотя бы одного нуля. Если он есть в этой части матрицы смежности, значит граф не полный, и, как следствие, размер клики уже меньше k $\Rightarrow$ слово не принадлежит языку ну или нам подсунули ложный сертификат.
Если же в этих строках и столбцах матрицы смежности нет нулей, значит между любыми двумя вершинами существует ребро и это действительно клика размера k $\Rightarrow$ слово принадлежит языку. Заметим, что проверка $k * k$ клеток матрицы смежности, где $k \leq n$ осуществляется за $O(n^2)$, значит этот язык $\mathbf{NP}$

4.2 Два графа являются изоморфными

Т.к. графы изоморфны, то существует биекция $V \Rightarrow V'$ и значит в качестве сертификата достаточно предъявить перестановку вершин, чтобы графы (и их матрицы смежности совпали). Графы A и B заданы их матрицами смежности a, b размера n^2, где n- количество вершин. Тогда для проверки мы просто за $O(poly(n^2)$ меняем местами строки и стобцы второй матрицы смежности в соответствии с перестановкой вершин и сравниваем ячейки в них. Если они полностью совпали - победа, после перестановки получился один и тот же граф, значит они были изоморфны



a	1	2	3	4
1	0	1	0	1
2	1	0	1	1
3	0	1	0	0
4	1	1	0	0



b	1	2	3	4
1	0	1	0	1
2	1	0	0	1
3	0	0	0	1
4	1	1	1	0

5 Построим сертификат простоты для числа $p = 3931, g = 2$.

Действуя по алгоритму с семинара:

1. $p - 1 = 2 * 3 * 5 * 131 = p_1^{k_1} * p_2^{k_2} * p_3^{k_3} * p_4^{k_4}$
   В соответствующие ячейки нашего дерева помещаем числа в таком формате: p_i, k_i, g_i. Т.е. корень, его кратность и первообразный корень. Первообразные корни мы находим просто: для 2 это 1, а для остальных простых чисел им может послужить 2 (т.к. по соответствующей теореме с семинара все степени по модулю различны и не равны нулю).

2. $p_1 = 2, p_2 = 3, p_3 = 5$ мы считаем простыми и на них рекурсия останавливается. Далее у нас есть $p_4 = 131$, для которого: $p -_4 -1 = 130 = 2 * 5 * 13$.

3. Далее $p_{4,3} - 1 = 13 - 1 = 2^2 * 3$

Проверяем сертификат снизу вверх за полиномиальное (доказано на семинаре) от длины время:

1. Действительно, $13 = 2^2 * 3 + 1$. Простоту 2,3,5 мы считаем известной.

2. $131 = 2 * 5 * 13 + 1$. Проверим первообразный корень g=2:

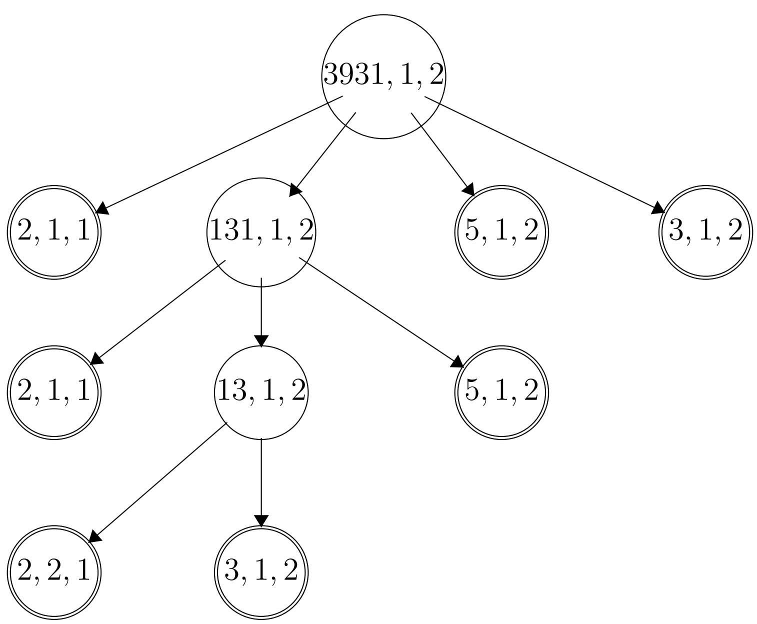
$$2^{\frac{13-1}{2}} = 64 \neq 1, \quad 2^{\frac{13-1}{3}} = 16 \neq 1$$

3. И правда, $3931 = 2 * 3 * 5 * 131 + 1$ и для первообразного корня $g = 2$ тоже всё хорошо:

$$2^{\frac{3931-1}{2}} = 3930 \neq 1; \quad 2^{\frac{3931-1}{3}} = 3930 \neq 3313; \quad 2^{\frac{3931-1}{5}} = 3250 \neq 1; \quad 2^{\frac{3931-1}{131}} = 967 \neq 1;$$

В итоге получили, что теорема о первообразных корнях выполняется, и число 3931 действительно простое.

Замечание: у нас всё хорошо с возведением в степень потому что при вычислении на больших числах можно это реализовать функцией $Modular - Exponentiation$, приведённой на стр. 1000 Кормена. Её ассимптотика $O(n^3)$, где n - число битов, которое занимает вход.



6 Доказать что класс несовместных систем линейных уравнений $\in$ **NP**

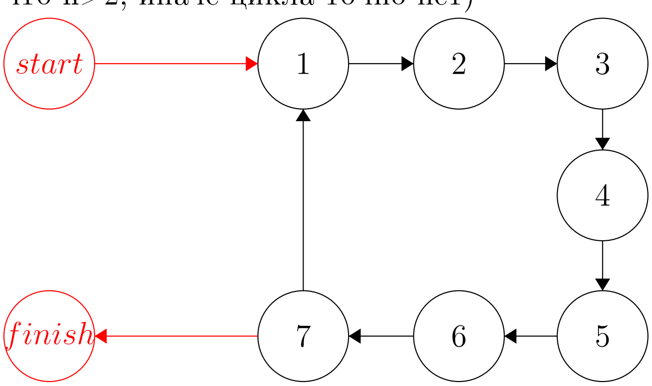
По теореме Фредгольма $Ax = b -$ совместна $\Leftrightarrow$ каждое решение сопряжённой однородной системы $c^T A = 0$ удовлетворяло уравнению $c^T b = 0$.
Значит мы можем в качестве сертификата предъявлять такое решение c_0 сопряжённой однородной системы, что $c_0^T b \neq 0$. Тогда проверка осуществляется подстановкой в расширенную матрицу системы $||A|B||$ решение c_0. И далее операциями над матрицей убеждаемся в нарушении теоремы Фредгольма.
Находить c_0 можно решая методом Гаусса сопряжённую однородную систему. Причём это происходит за полином т.к. каждый элемент c_0 - полином от элементов исходной матрицы, степени не выше 2018 (из определения детерминанта). И такой проблемы, как на семинаре у нас не будет, ведь размеры миноров ограничены 2018 (что есть константа).
Значит мы предъявили сертификат и алгоритм проверки, за полиномиальное время. Это и доказывает принадлежность языка **NP**

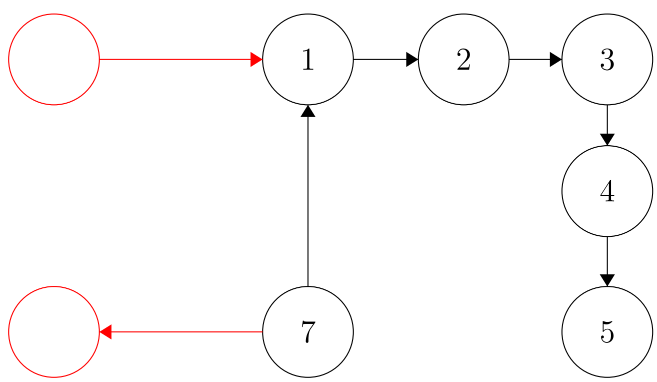
7 Построить явные полиномиальные сводимости языков графов грамильтоновых путей и циклов

*для краткости под циклом и путём тут подразумеваются гамильтоновы цикл и путь.

7.1 Из цикла в путь

Функция сводимости - дорисовывание висячих рёбер к двум **смежным** вершинам (считаем, что n>2, иначе цикла точно нет)





Если у нас был цикл, то при добавлении *start* и *finish* т.к. они висячие мы обязаны начать и закончить путь в них, иначе они просто не будут посещены. Например, из цикла $1 \to 2 \to 3 ... \to 7 \to 1$ получим путь $start \to 1 \to 2 ... \to 7 \to finish$

Если же цикла не было, то при добавлении двух висячих вершин каким-то (выбираем любые) двум смежным (например 1 и 7) мы всё ещё обязаны в одной начать (start), а в другой закончить (finish), но поскольку цикла не было, то путь $start \to 1 \to 7 \to finish$ не будет гамильтоновым (2,3,4,5 не посещены)

Заметим, что две связанные вершины находятся за полином - пробежаться по матрице смежности, например, или поиском в глубину. Таким образом мы доказали корректность и полиномиальность функции сводимости от цикла к пути (добавление висячих рёбер к двум смежным вершинам).

7.2 От пути к циклу

Преобразование: добавим новую вершину a, и рёбра из неё во все вершины графа. Тогда:

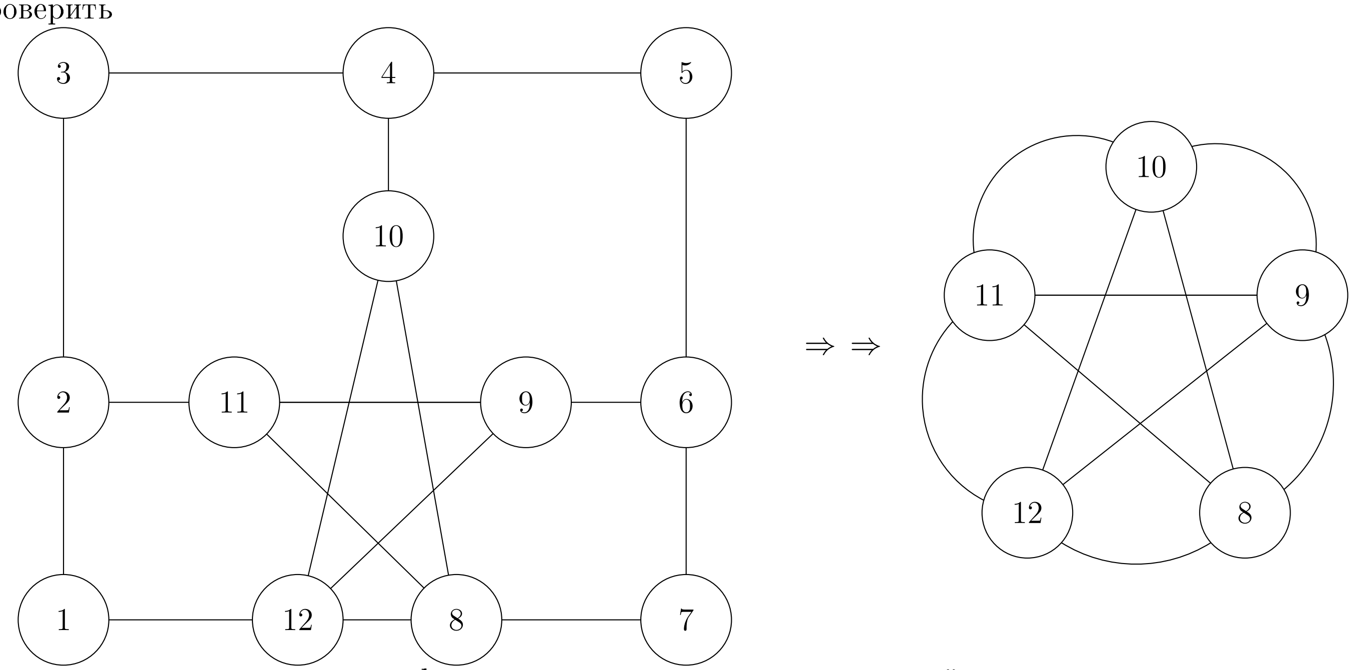
1. Если путь был, значит он имел вид: $s \to \ldots \to f$, но теперь можно из f дойти до s через a. Получим цикл: $s \to \ldots \to f \to a \to s$

2. Докажем, что если пути не было, то и цикл не появится. Допустим обратное и у нас нет пути, но при добавлении a появился цикл вида: $s \to \ldots \to s_k \to a \to s_{k+1} \to \ldots \to s_m \to s$. Но тогда если убрать вершину a и все рёбра из неё, у нас останется путь: $s_{k+1} \to \ldots \to s_m \to s \to \ldots \to s_k$. Противоречие, значит цикл не появится, если пути не было.

Заметим, что добавление вершины и ребёр во все остальные тоже $O(n)$ т.е. полиномиально от длины входа (n^2). Корректность мы уже доказали.

8 Показать, что $L_{planar} \in$ coNP

По определению **coNP** условие задачи сводится к "покажите, что язык непланарных графов является **NP** " . Вспомним критерий-теорему Понтрягина-Куратовского, которая говорит "если в графе существует подграф, гомеоморфный K_5 или $K_{3,3}$, то граф не планарный".
Используя эту теорему мы можем в качестве сертификата давать на вход список "схлопываний"вершин, чтобы получить подграф, гомеоморфный K_5 или $K_{3,3}$. Например, для графа слева сертификатом будет $y = (1, 12), (2, 11), (3, 4), (5, 4), (4, 10), (6, 9), (7, 8); [8, 9, 10, 11, 12]$. Т.е. сначала мы подаём вершины попарно (какую, куда схлопнуть), а затем [список вершин подграфа, являющегося K_5 или $K_{3,3}$]. Получилось, что данный пример сводится к K_5 и сертификат помогает нам это проверить

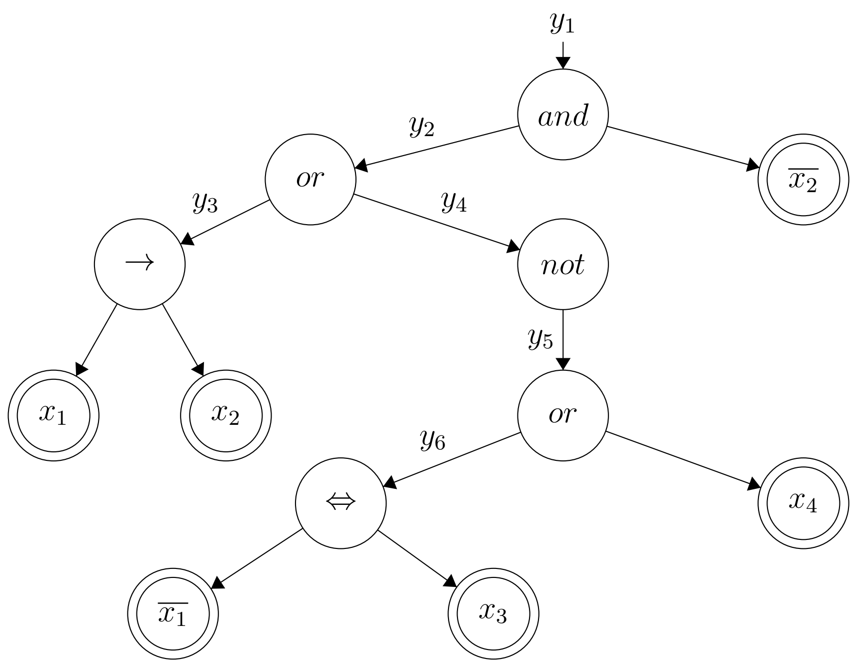


Заметим, что если нам дан граф из n вершин, то длина входа задаётся матрицей смежности (n^2). И поскольку схлопываний не больше, чем n (за 1 итерацию избавляемся от одной вершины), то длина сертификата $O(n)$, а сложность проверки $O(n^3) + const$ - ведь схлопывая вершины, мы изменяем матрицу смежности. Ну а сама проверка 5-6 вершин на K_5 или $K_{3,3}$ вообще константное время. Таким образом мы доказали, что $L_{planar} \in$ **coNP**

9 Доказать что $EXACT - 3SAT \in \mathbf{NP}$

*решение задачи честно найдено в Кормене (теорема 34.10 на стр. 1131 там же и конкретный пример работы), а затем понято и осознано.
Сам алгоритм сведения $3SAT \leq_P 3 - CNF - SAT$ проходи в два основных этапа:

1. Для формулы ϕ строится бинарное синтаксическое дерево, листья которого - литералы, а узлы - бинарные (или унарные) операции. Вот дерево для $\phi = \Big((x_1 \to x_2)\ or\ \overline{((\overline{x_1} \leftrightarrow x_3) or\ x_4)}\Big) or\ \overline{x_2}$.



   Дерево рассматриваем как схему для вычисления функции. Для каждого внутреннего узла вводим переменные y_i, и переписываем формулу как конъюнкцию переменной корня и выражений операций в каждом узле. Т.е. в итоге мы получим формулу ϕ_1, являющуюся конъюнкцией выражений из не более чем трёх литералов. Но каждое выражение в скобках (ϕ_{1i}) должно быть дизъюнкцией ровно трёх литералов.

2. Приведём ϕ_1 к КНФ. По таблице истинности формул подвыражений будем из строк с нулевым результатом запишем $\overline{\phi_{1i}}$ в ДНФ, а затем по законам де Моргана приведём эту формулу в КНФ-формулу ϕ_{2i}.

bonus Если вдруг условие "ровно 3 литерала в каждом дизъюнкте" нарушилось (оказалось меньше трёх), то мы просто создаём формулу ϕ_3 по формуле ϕ_2 так: если литерала два, то добавляем новый литерал p и дописывам в формулу подвыражение $(l_1\ or\ l_2\ or\ p)\ and\ (l1\ or\ l2\ or\ \overline{p})$. Аналогично, но уже с двумя литералами p и q, если в подформуле был только один литера, дописываем в итоговую КНФ 4 формулы (l_1 со всеми комбинациями $p, q, \overline{p}, \overline{q}$. Если литерала три, то просто переписываем в ϕ_3 скобку неизменно

Теперь по ассимптотикам: от ϕ к ϕ_1 добавляем не больше n переменных и n подвыражений (n-начальное количество литералов). При построении ϕ_2 из ϕ_1 добавляется не больше 8ми строк (т.к.таблица истинности состоит из 2^3 строк). Для ϕ_3 по ϕ_2 в худшем случае добавляется по 4 подвыражения для каждого подвыражения ϕ_2. Поэтому длина сертификата $O(n)$.