АМВ. ДЗ на неделю 3.

Прохоров Юрий, 771

Примеры $\mathcal{NP}$-полных языков

- **CIRCUIT-SAT**
  Выполнимые булевы схемы (SAT — satisfiability problem).
- **SAT** (ВЫПОЛНИМОСТЬ)
  Выполнимые булевы формулы, содержащие операции конъюнкции, дизъюнкции и отрицания $(\wedge, \vee, \neg)$.
- **3-CNF SAT** (3-ВЫПОЛНИМОСТЬ)
  Выполнимые КНФ, в которых в каждый дизъюнкт входит не более трех литералов, причем литералы в одном дизъюнкте отвечают различным логическим переменным (язык, в котором литералы в одном дизъюнкте могут соответствовать одним и тем же переменным также $\mathcal{NP}$-полный).
- **РОВНО-3-ВЫПОЛНИМОСТЬ**
  То же самое, что и 3-ВЫПОЛНИМОСТЬ, но в каждом дизъюнкте ровно 3 литерала. В одном дизъюнкте так же могут как быть, так и не быть одинаковые переменные.
- **MAX-2-ВЫПОЛНИМОСТЬ**
  2-КНФ (КНФ, в каждый дизъюнкт которой входит не более двух литералов) и число k такие, что существует набор значений, при котором истинны хотя бы k дизъюнктов.
- **NAE-SAT**
  Not-All-Equal-SAT. КНФ-формулы, для которых существует такой набор значений, что в каждом дизъюнкте есть как истинный, так и ложный литерал.
- **CLIQUE** (КЛИКА)
  Граф G и число k такие, что в графе G есть полный подграф (клика) на k вершинах.
- **VERTEX-COVER** (ВЕРШИННОЕ ПОКРЫТИЕ)
  Граф G и число k такие, что существует такое множество из k вершин графа G, что хотя бы один конец любого ребра графа G лежит в этом множестве.
- **HAMPATH** (ГАМИЛЬТОНОВ ПУТЬ, ГП)
  Графы, в которых есть гамильтонов путь.
- **HAMCYCLE** (ГАМИЛЬТОНОВ ГРАФ, ГАМИЛЬТОНОВ ЦИКЛ, ГЦ)
  Графы, в которых есть гамильтонов цикл.
- **MAX-CUT** (МАКСИМАЛЬНЫЙ РАЗРЕЗ)
  Граф G и число k такие, что множество вершин графа можно разбить на 2 непересекающихся множества, между которыми можно провести не менее k ребер.
  Иногда рассматривается взвешенный вариант задачи: граф $G = (V, E)$, неотрицательная весовая функция на ребрах $w : E \longrightarrow \mathbb{Z}_+$ и число k такие, что вершины графа можно разбить на 2 непересекающихся множества V_1 и V_2 так, что сумма весов всех ребер между V_1 и V_2 не менее k.
- **CHROMATIC NUMBER** (ХРОМАТИЧЕСКОЕ ЧИСЛО)
  Неориентированный граф G и число k такие, что вершины графа G можно покрасить в k цветов так, чтобы любые две смежные вершины были разных цветов.
- **3-COLOUR**
  Графы, для которых существует корректная раскраска вершин в 3 цвета (см. выше).
- **ПРОТЫКАЮЩЕЕ МНОЖЕСТВО**
  Семейство конечных множеств $\{A_1, A_2, \ldots, A_m\}$ и число k такие, что существует множество мощности k, имеющее непустое пересечение с каждым из множеств A_i $(i = 1, 2, \ldots, m)$.

- **3 DIMENSIONAL MATCHING** (3-СОЧЕТАНИЕ)
  Множество $M \subseteq X \times Y \times Z$, где X, Y, Z — непересекающиеся множества, и число k такие, что существует подмножество $M' \subseteq M$ мощности k такое, что у ни у каких двух его элементов нет ни одной одинаковой координаты.
- **PARTITION** (РАЗБИЕНИЕ, ЗАДАЧА О КАМНЯХ)
  Конечное множество натуральных чисел такое, что его можно разбить на 2 подмножества так, что сумма чисел в подмножествах одинакова.
- **KNAPSACK** (РЮКЗАК)
  Конечное множество натуральных чисел A и число b такие, что существует подмножество $A' \subseteq A$ такое, что сумма его элементов равна b.

В задачах будем называть *логическими переменными* символы $x_1, x_2, \ldots, x_n$, а *литералами* — логические переменные или их отрицания.

Задача 1

Постройте полиномиальную сводимость языка 3-ВЫПОЛНИМОСТЬ (3-SAT) (выполнимые КНФ, в каждом дизъюнкте не более 3 литералов) к языку РОВНО-3-ВЫПОЛНИМОСТЬ (выполнимые КНФ, в каждом дизъюнкте в точности 3 литерала).

Решение:

Считаем в первом языке в дизъюнкте могут повторяться переменные, а во втором языке все переменные в дизъюнкте различны.

Из булевой алгебры известны тождества

$$z \vee \neg z = 1, \qquad z \vee z = z, \qquad \neg z \vee \neg z = \neg z, \qquad \forall z \in \{0, 1\},$$

поэтому с помощью этих эквивалентных преобразований можно избавиться от повторяющихся переменных в дизъюнкте.

Заметим, что для некоторых переменных u, v, w:

$$\neg u \wedge \neg v \wedge \neg w = (\neg u \vee \neg v \vee \neg w) \wedge (\neg u \vee \neg v \vee w) \wedge \ldots \wedge (u \vee v \vee \neg w)$$

То есть конъюнкцию отрицаний трех переменных можно представить в виде полной КНФ из 7 членов.

Пусть u, v, w — переменные, которых нет среди переменных $x_1, x_2, \ldots, x_n$ из исходной КНФ. Опишем действия сводящей функции f, преобразовывающей 3-КНФ в РОВНО-3-КНФ:

$$B = f(A)$$

1. Удалить повторяющиеся переменные в дизъюнктах.
2. Дизъюнкты с одним литералом (a_i) заменить на дизъюнкты с тремя литералами $(a_i \vee u \vee v)$.
3. Дизъюнкты с двумя литералами $(a_i \vee a_j)$ заменить на дизъюнкты с тремя литералами $(a_i \vee a_j \vee u)$.
4. Дописать к КНФ еще 7 дизъюнктов из полной КНФ для $(\neg u \wedge \neg v \wedge \neg w)$.

Преобразование f выполняется, как видно из описания, за полиномиальное от длины записи КНФ время. Докажем, что выполнимость $B = f(A)$ эквивалентна выполнимости A.

Допустим, A выполнима. Тогда существует такой набор значений $\alpha = (\alpha_1, \ldots, \alpha_n)$, что в каждом дизъюнкте в A есть литерал, равный 1. Рассмотрим набор $\beta = (\alpha_1, \ldots, \alpha_n, 0, 0, 0)$ для переменных $x_1, \ldots, x_n, u, v, w$. Последние 7 дизъюнктов будут истинными, так как их совокупность эквивалентна $(\neg u \wedge \neg v \wedge \neg w)$, что равно 1 при нулевых переменных. А все оставшиеся дизъюнкты будут истинны, так как α — выполняющий набор для A.

Допустим, B выполнима. Тогда выполнима полная КНФ из 7 последних дизъюнктов B, значит, $u = v = w = 0$. Тогда выполняющий набор имеет вид $\beta = (\alpha_1, \ldots, \alpha_n, 0, 0, 0)$. Подставим u, v, w как нули в B. Получим КНФ, которая получается в преобразовании f после первого шага и для нее есть выполняющий набор β. Но эта КНФ эквивалентна формуле A, поэтому сама A выполнима набором $\alpha = (\alpha_1, \ldots, \alpha_n)$.

Задача 2

Постройте сводимость языка ВЫПОЛНИМОСТЬ (SAT) к языку ПРОТЫКАЮЩЕЕ МНОЖЕСТВО. Перед этим проделайте пункты (i) и (ii) задачи 34 для иллюстрации и понимания происходящего.

Решение:

Преобразование f формулы ϕ в семейство множеств A_ϕ и число k:

1. Построить по формуле булеву схему за полиномиальное время (обсуждали на семинаре).
2. Построить по булевой схеме 3-КНФ с помощью алгоритма из задачи 7 (его также разбирали на семинаре).
3. Строим семейство множеств A_ϕ. Пусть 3-КНФ имеет n переменных и m дизъюнктов.
   - Добавляем n множеств вида $\{x_i, \neg x_i\},\ i = 1, 2, \ldots, n$
   - Добавляем m множеств $\{a_j, b_j, c_j\}$ для дизъюнктов $(a_j \vee b_j \vee c_j),\ j = 1, 2, \ldots, m$
4. $k = n$ — число переменных в ϕ.

Все шаги требуют полиномиального времени. Докажем, что для A_ϕ есть k-элементное протыкающее множество тогда и только тогда, когда формула ϕ выполнима. Корректность шага 1 обсуждалась на семинаре, корректность шага 2 будет доказана в задаче 7. Докажем корректность шага 3, то есть формулу ϕ считаем 3-КНФ.

Пусть ϕ выполнима. Тогда существует выполняющий набор $\alpha = (\alpha_1, \ldots, \alpha_n)$. Ему соответствует набор литералов $a = (x_1^{\alpha_1}, \ldots, x_n^{\alpha_n})$, где $y^1 = y,\ y^0 = \neg y$. Набор α — выполняющий, значит, в каждом из m множеств, соответствующих дизъюнктам, есть какой-то литерал из набора a. Кроме того, в наборе a есть литералы, соответствующие каждой из n переменных, поэтому он пересекает первые n множеств A_ϕ. Таким образом, набор a является протыкающим множеством из n элементов.

Пусть существует протыкающее множество из n элементов для A_ϕ. Тогда для каждой из n переменных есть ровно один литерал, соответствующий ей, так как нужно покрыть первые n множеств. Поэтому это протыкающее множество имеет вид $a = (x_1^{\alpha_1}, \ldots, x_n^{\alpha_n})$. a — протыкающее множество, поэтому в каждом из последних m множествах A_ϕ есть какой-то элемент набора a. Но те множества были построены по дизъюнктам, поэтому при наборе значений $\alpha = (\alpha_1, \ldots, \alpha_n)$ в каждом дизъюнкте будет хотя бы одна единица, значит, ϕ выполнима.

(i) $\psi = x_1 \vee x_2 \vee \neg x_3$

В обозначениях из доказательства:

$$\alpha = (1, 1, 1) \qquad \Longrightarrow \qquad a = (x_1, x_2, x_3)$$

(ii) $\chi = (x_1 \vee x_2) \wedge (x_1 \vee \neg x_2) \wedge \neg x_1$

По методу резолюций из ТФС, следующие формулы равносильны с точки зрения выполнимости:

$$\chi \iff x_1 \wedge \neg x_1$$

Но противоречие невыполнимо, следовательно, χ невыполнима. По доказанному общему утверждению, A_χ не имеет протыкающего множества из 2 элементов и менее.

Задача 3

Постройте сводимость языка 3-ВЫПОЛНИМОСТЬ к языку ВЕРШИННОЕ ПОКРЫТИЕ (VERTEX-COVER). Перед этим проделайте пункты (i) и (ii) задачи 35 для иллюстрации и понимания происходящего.

Решение:

В связи со сводимостью, доказанной в задаче 1, считаем что язык 3-ВЫПОЛНИМОСТЬ состоит из выполнимых РОВНО-3-КНФ с различными переменными в каждом из дизъюнктов. Также далее считаем, что формула ϕ имеет n переменных и m дизъюнктов.

Преобразование f формулы ϕ в граф G_ϕ и число k:

1. Создаем граф G_ϕ из $2n + 3m$ вершин:

   - Добавляем $2n$ вершин, помеченных x_i и $\neg x_i$ и соединяем ребрами вершины с одинаковыми индексами. Все эти вершины назовем *литеральными.* Ребра между ними тоже назовем *литеральными.*
   - Добавляем $3m$ вершин, помеченных $j{-}a_j, j{-}b_j, j{-}c_j$, где a_j, b_j, c_j — литералы из j-го дизъюнкта $(a_j \vee b_j \vee c_j)$. Попарно соединяем вершины каждой j-ой тройки. Все эти вершины назовем *дизъюнктными* и аналогично ребра между ними.
   - $\forall j = 1, 2, \ldots, m$:
     Соединяем $j - a_j, j - b_j, j - c_j$ с литеральными вершинами a_j, b_j, c_j соответственно. Эти ребра назовем *промежуточными* и их ровно 3 для каждой тройки дизъюнктных вершин.

2. $k = n + 2m$.

Видно, что все шаги занимают полиномиальное время. Докажем, что 3-КНФ ϕ выполнима тогда и только тогда, когда в графе G_ϕ есть вершинное покрытие из $k = n + 2m$ вершин.

Пусть ϕ выполнима. Тогда существует набор значений α и соответствующий ему набор литералов a (см. задачу 2), который на этом наборе обращается в единичный вектор. Начнем строить покрытие V. Добавим в V набор a, так мы покрыли все литеральные ребра. Кроме того, по построению, в каждой из троек дизъюнктных вершин есть одна, соединенная с какой-то из набора a, потому что КНФ выполнима. Таким образом, для каждой тройки дизъюнктных вершин мы покрыли хотя бы одно промежуточное ребро.

Осталось покрыть все дизъюнктные ребра и оставшиеся 2 промежуточных ребра (или, быть может, даже меньше) для каждой тройки дизъюнктных вершин. Из каждой тройки ($\forall j = 1, \ldots, m$) добавим в V те две вершины, которые покроют оставшиеся 2 промежуточных ребра (если осталось меньше двух, то выбор произвольный). Так мы покрыли все промежуточные ребра. Заметим, что из каждой тройки дизъюнктных вершин мы взяли по 2 вершины, значит, мы покрыли все дизъюнктные ребра тоже. Итак, V — вершинное покрытие размера $k = n + 2m$.

Пусть в G_ϕ есть вершинное покрытие V размера $k = n + 2m$. Литеральные ребра не имеет общих вершин между собой, значит, на каждое из n литеральных ребер нужно по одной вершине. Следовательно, в V есть литералы для каждой переменной. Дизъюнктные ребра образуют m треугольников, и треугольники не имеют общих вершин между собой и с литеральными ребрами. Значит, на каждый из m треугольников нужно по две вершине. Так мы поняли, как распределены все k вершин покрытия V между вершинами разных типов.

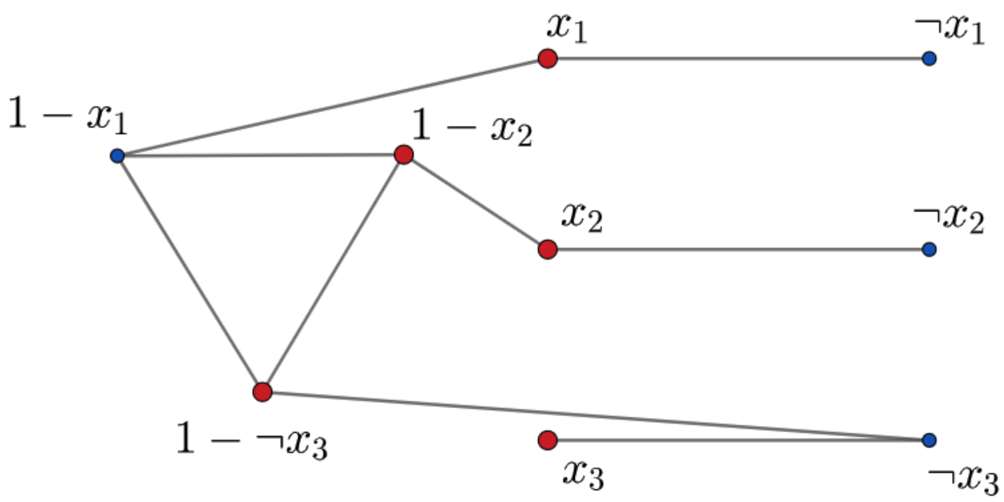
V является покрытием, значит, все промежуточные ребра покрыты. Рассмотрим произвольную j-ую тройку дизъюнктных вершин. В покрытие V входят ровно две из трех вершин, значит, ими покрыты только 2 промежуточных ребра. Тогда третье ребро должно быть покрыто некоторой литеральной вершиной, попавшей в V. Итак, каждая тройка соединена некоторым промежуточным ребром с какой-то литеральной вершиной из V. По построению, это означает в в каждый дизъюнкт входит какой-то литерал из V.

Таким образом, мы показали, что литеральные вершины вершины из V (их всего n) образуют такой набор a, что в каждом дизъюнкте есть элемент из a. Кроме того, мы показали, что в a есть литералы, соответствующие каждой из переменных. Тогда набор имеет вид $a = (x_1^{\alpha_1}, \ldots, x_n^{\alpha_n})$, а набор значений $\alpha = (\alpha_1, \ldots, \alpha_n)$ является выполняющим для исходной КНФ.

(i) $\psi = x_1 \vee x_2 \vee \neg x_3$

Выполняющий набор:

$$\alpha = (1, 1, 1) \qquad \Longrightarrow \qquad a = (x_1, x_2, x_3)$$



Покрытие V из 5 вершин выделено на рисунке красным цветом.

(ii) $\chi = (x_1 \vee x_2) \wedge (x_1 \vee \neg x_2) \wedge \neg x_1$

Формула χ невыполнима, значит, по доказанному выше более общему утверждению, в графе G_χ не существует вершинного покрытия из $k = 8$ или менее вершин.

Задача 4

Постройте сводимость языка РОВНО-3-ВЫПОЛНИМОСТЬ к языку КЛИКА (CLIQUE). Перед этим проделайте пункты (i) и (ii) задачи 36 для иллюстрации и понимания происходящего.

Решение:

В связи со сводимостью, доказанной в задаче 1, считаем что язык РОВНО-3-ВЫПОЛНИМОСТЬ состоит из выполнимых РОВНО-3-КНФ с различными переменными в каждом из дизъюнктов. Также далее считаем, что формула ϕ имеет n переменных и m дизъюнктов.

Преобразование f формулы ϕ в граф G_ϕ и число k:

1. Создаем граф G_ϕ из $3m$ вершин:
   - Для каждого дизъюнкта $(a_j \vee b_j \vee c_j)$, $j = 1, \ldots, m$, создаем тройку вершин $j - a_j, j - b_j, j - c_j$.
   - Для любых двух вершин из разных троек $i - a_i$ и $j - b_j$ $(i \neq j)$ соединяем их, если $a_i \neq \neg b_j$ как литералы.
2. $k = m$.

Видно, что все шаги занимают полиномиальное время. Докажем, что КНФ ϕ выполнима тогда и только тогда, когда в графе G_ϕ есть клика размера $k = m$.

Пусть КНФ ϕ выполнима. Как и в предыдущих задачах, существует выполняющий набор α и соответствующий набор литералов a. В каждом дизъюнкте есть какой-то литерал v_j из a. Обозначим набор вершин графа, соответствующих этим литералам, $V = \{1 - v_1, 2 - v_2, \ldots, m - v_m\}$. Покажем, что в G_ϕ есть клика на этих вершинах.

Возьмем произвольные вершины $i - v_i$ и $j - v_j$, $i \neq j$. Пусть литералы v_i и v_j отвечают разным логическим переменным. Тогда, по построению, они соеденены в графе, так как они из разных троек. Пусть v_i и v_j

отвечают одной и той же переменной. Но $v_i, v_j \in a$, а в наборе a не более одного литерала для каждой переменной, значит, $v_i = v_j$. По построению, они будут соединены в графе G_ϕ. Итак, любые две вершины попарно соединены, значит, этот подграф является кликой.

Пусть G_ϕ есть клика размера m. Так как внутри каждой из m троек никакие две вершины не соединены, то в каждой тройке будет ровно по одной вершине. То есть множество вершин клики имеет вид $V = \left\{1 - v_1, 2 - v_2, \dots, m - v_m\right\}$.
По построению, среди литералов, соответствующим вершинам клики, нет таких v_i и v_j, что $v_i = \neg v_j$. Тогда для каждой логической переменной $x_1, \dots, x_n$ в V существует не более одного различного литерала. Рассмотрим набор $\tilde{a}$ всех различных литералов, соответствующих вершинам клики V. Тогда, по построению, в любом дизъюнкте (тройки соответствуют дизъюнктам) есть литерал из набора $\tilde{a}$. Это набор можно произвольным образом дополнить литералами, соответствующими оставшимся логическим переменным, до набора a. Этот набор и определяет выполняющий набор значений α для исходной КНФ.

(i) $\psi = x_1 \vee x_2 \vee \neg x_3$

Граф G_ψ состоит из 3 вершин, и не имеет ребер. Кликой размера 1 будет любая вершина этого графа.

(ii) $\chi = (x_1 \vee x_2) \wedge (x_1 \vee \neg x_2) \wedge \neg x_1$

Формула χ невыполнима, значит, по доказанному выше более общему утверждению, в графе G_χ не существует клики размера $k = 3$ или более.

Задача 5

Постройте сводимость языка 3-ВЫПОЛНИМОСТЬ к языку max-2-ВЫПОЛНИМОСТЬ (max-2-SAT). Перед этим проделайте пункты (i) и (ii) задачи 37 для иллюстрации и понимания происходящего.

Решение:

В связи со сводимостью, доказанной в задаче 1, считаем что язык 3-ВЫПОЛНИМОСТЬ состоит из выполнимых РОВНО-3-КНФ с различными переменными в каждом из дизъюнктов. Также далее считаем, что формула ϕ имеет n переменных и m дизъюнктов.

Преобразование f формулы ϕ в формулу (2-КНФ) Φ и число k:

1. Строим 2-КНФ Φ, имеющую $10m$ дизъюнктов:
   - Для каждого дизъюнкта $(a_j \vee b_j \vee c_j)$, $j = 1, \dots, m$, добавляем в Φ следующие 10 дизъюнктов:
   $$\left\{a_j,\ b_j,\ c_j,\ d_j,\ \neg a_j \vee \neg b_j,\ \neg a_j \vee \neg c_j,\ \neg b_j \vee \neg c_j,\ a_j \vee \neg d_j,\ b_j \vee \neg d_j,\ c_j \vee \neg d_j\right\},$$
   где $d_1, d_2, \dots, d_m$ — m новых переменных.
2. $k = 7m$.

Видно, что все шаги занимают полиномиальное время. Докажем, что КНФ ϕ выполнима тогда и только тогда, когда существует такой набор значений, при котором истинны хотя бы $k = 7m$ дизъюнктов формулы Φ.

Пусть ϕ выполнима. Как и ранее, существует выполняющий набор $\alpha = (\alpha_1, \dots, \alpha_n)$ и набор литералов a. В каждом дизъюнкте есть хотя бы 1 литерал из набора a. Построим набор $\beta = (\alpha_1, \dots, \alpha_n, \beta_1, \dots, \beta_m)$ для переменных $x_1, \dots, x_n, d_1, \dots, d_m$, от которых зависит Φ. Рассмотрим j-ый дизъюнкт. Исходя из таблицы ниже, в качестве $d_j = \beta_j$ выбираем 0 или 1 в зависимости от того, сколько литералов дизъюнкта входит в набор a (если один или два — $\beta_j = 0$, если три — $\beta_j = 1$, ноль быть, ясно, не может). Тогда при $x = \alpha$ и $d_j = \beta_j$ ровно 7 2-дизъюнктов десятки, соответствующей этому 3-дизъюнкту, будут истинными. Это верно для любого дизъюнкта, а значения β_j не зависят друг от друга. Поэтому в конечной 2-КНФ при наборе β будет истинно ровно $7m$ дизъюнктов.

a_j	b_j	c_j	d_j	$\neg a_j \vee \neg b_j$	$\neg a_j \vee \neg c_j$	$\neg b_j \vee \neg c_j$	$a_j \vee \neg d_j$	$b_j \vee \neg d_j$	$c_j \vee \neg d_j$	# истин
0	0	0	0	1	1	1	1	1	1	6
0	0	0	1	1	1	1	0	0	0	4
1	0	0	0	1	1	1	1	1	1	7
1	0	0	1	1	1	1	1	0	0	6
1	1	0	0	0	1	1	1	1	1	7
1	1	0	1	0	1	1	1	1	0	7
1	1	1	0	0	0	0	1	1	1	6
1	1	1	1	0	0	0	1	1	1	7

Пусть в Φ выполнимо $7m$ дизъюнктов при наборе $\beta = (\alpha_1, \ldots, \alpha_n, \beta_1, \ldots, \beta_m)$. Из таблицы выше видно, что в каждой десятке должно быть выполнено по 7 2-дизъюнктов и что это возможно, когда $\forall j$ среди a_j, b_j, c_j есть хотя бы одна единица при данном наборе β. Тогда в каждом 3-дизъюнкте исходной КНФ ϕ будет единица при наборе $\alpha = (\alpha_1, \ldots, \alpha_n)$. Тогда ϕ выполнима.

(i), (ii) $\psi = x_1 \vee x_2 \vee \neg x_3$

$f(\psi) = \big(x_1\big) \wedge \big(x_2\big) \wedge \big(\neg x_3\big) \wedge \big(d_1\big) \wedge \big(\neg x_1 \vee \neg x_2\big) \wedge \big(\neg x_1 \vee x_3\big) \wedge \big(\neg x_2 \vee x_3\big) \wedge \big(x_1 \vee \neg d_1\big) \wedge \big(x_2 \vee \neg d_1\big) \wedge \big(\neg x_3 \vee \neg d_1\big)$

ψ — выполнима, значит в $f(\psi)$ выполнимо $k = 7$ дизъюнктов.

Действуя по алгоритму из доказательства, имеем

$$\alpha = (1, 1, 1) \qquad \Longrightarrow \qquad a = (x_1, x_2, x_3)$$

В дизъюнкт входят 2 литерала из набора a, поэтому $\beta_1 = 0$ (но можно и 1, из таблицы видно, что это неважно). Итак, имеем набор

$$\beta = (1, 1, 1, 0).$$

Задача 6

Покажите, что если $3 - COLOUR$ лежит в $\mathcal{P}$, то за полиномиальное время можно не только определить, допускает ли граф раскраску, но и найти саму эту раскраску (если она есть). Обратите внимание, что на вход проверяющей 3-раскрашиваемость процедуры нельзя подавать частично окрашенные графы, спрашивая, можно ли дораскрасить оставшиеся вершины до полной правильной раскраски.

Решение:

Пусть $C(G)$ — функция, которая проверяет принадлежность $G \in 3 - COLOUR$ за полиномиальное время. Пусть G_0 — граф, для которого требуется построить раскаску. В качестве $[u, v] \in G_0$ будем обозначать ребро графа G_0, соединяющее вершины u и v.

Основной идеей алгоритма, является вычисление за полиномиальное время функции f, определенной на вершинах графа G_0:

$$f(u, v) = \begin{cases} 0, & u \text{ и } v \text{ разного цвета} \\ 1, & \text{иначе} \end{cases}$$

Мы покажем, что f будет задавать некоторую корректную раскраску.

Алгоритм построения раскраски графа G_0:

1. Проверить, что $C(G_0) = 1$, т.е. что раскраска существует.

2. Вычислить функцию f:

- Положить соседей разного цвета:
  $\forall [u, v] \in G_0 : \quad f(u, v) := 0$
- Создать промежуточный граф $G = G_0$.
- $\forall u, v$ (u, v — не соседи в исходном графе G_0):
  - Добавить ребро $[u, v]$ в G:
    $G = G + [u, v]$
  - Проверить, осталась ли раскраска:

    if $C(G) = 0$ then
        $G = G - [u, v]$, удаляем это ребро, если раскраска пропала
        $f(u, v) = 1$, полагаем эти вершины одного цвета
    else
        $f(u, v) = 0$

3. Раскрасить граф G_0 в цвета $\{A, B, C\}$:
   - Выбрать произвольную вершину s и раскрасить ее в цвет A.
   - Перебрать все вершины G_0 и найти такие, что $f(s, v) = 1$, раскрасить все такие вершины v в цвет A.
   - Выбрать любую непокрашенную вершину t и раскрасить ее в цвет B.
   - Перебрать все вершины G_0 и найти такие, что $f(t, u) = 1$, раскрасить все такие вершины u в цвет B.
   - Все непокрашенные вершины покрасить в цвет C.

Пусть время работы функции $C(G)$ есть $O(n^p)$. Асимптотическое время работы алгоритма определяется временем работы шага 2, на который нужно время (n — число вершин графа):

$$T(n) = O\Big(n^2 \cdot n^p\Big) = O\Big(n^{p+2}\Big)$$

Будем считать, что бинарное отношение f является рефлексивным и коммутативным, т.е.

$$\forall u, v : \qquad f(u, u) = 1, \qquad f(u, v) = f(v, u)$$

Лемма 1. *Пусть Col — некоторая корректная раскраска графа G в k цветов (отображение множества вершин в k-элементное множество). Пусть G_0 — подграф G. Тогда сужение Col на множество вершин G_0 — корректная раскраска графа G_0.*

Доказательство:
Допустим противное, существует такая пара вершин $u, v \in G_0$, что

$$[u, v] \in G_0 \qquad \text{и} \qquad Col(u) = Col(v)$$

Так как G_0 — подграф G, то эта же пара вершин будет противоречить корректности раскраски Col для графа G. □

Лемма 2. *Пусть $R(u, v)$ — некоторое бинарное отношение на вершинах графа. Тогда R задает раскраску (1, если вершины одного цвета, 0 — иначе) графа в k цветов (необязательно корректную) тогда и только тогда, когда R является отношением эквивалентности и имеет не более k классов эквивалентности.*

Доказательство:
Достаточность. Пусть R — отношение эквивалентности с $m \leq k$ классами. Так как классы эквивалентности являются дизъюнктным разбиением множества вершин, то вершины каждого из классов красим в свой цвет. Классов $m \leq k$, значит, цветов хватит.

Необходимость. Пусть для раскраски вершин графа в k цветов задано бинарное отношение R. По определению раскраски ясно, что R является рефлексивным, коммутативным и транзитивным отношением.

Тогда R — отношение эквивалентности. Количество классов при этом равно числу цветов k. $\square$

Для обоснования корректности шага 2 нам надо доказать, что вычисленное в алгоритме бинарное отношение f является транзитивным и что оно разбивает множество вершин на не более, чем 3 класса эквивалентости. Так мы покажем, что f действительно задает какую-то раскраску в 3 цвета, а не является произвольной функцией.

Лемма 3. *Получаемое в результате работы алгоритма бинарное отношение f является транзитивным.*

Доказательство:
Зафиксируем произвольный порядок обхода пар несоседних вершин графа G_0 во втором шаге алгоритма. Пусть в некоторый момент рассматриваются вершины v и w, и к этому моменту уже вычислено, что $f(u,v) = f(u,w) = 1$. Заметим, что с каждой итерацией цикла в графе G только увеличивается количество ребер, поэтому на данном шаге $C(G + [u,v]) = C(G + [u,w]) = 0$, это следует из леммы 1.

Так как $C(G + [u,v]) = 0$, то в любой корректной раскраске текущего графа G вершины u и v имеют одинаковый цвет. Аналогично с вершинами u и w. Тогда в любой раскраске G вершины v и w имеют одинаковый цвет, поэтому при добавлении ребра $[v,w]$ раскраска пропадет, и $f(v,w) = 1$. $\square$

Лемма 4. *Получаемое в результате работы алгоритма отношение эквивалентности f разбивает вершины графа на не более, чем 3 класса эквивалентности.*

Доказательство:
Заметим, что на каждой итерации цикла в шаге 2 алгоритма поддерживается свойство $C(G) = 1$. Кроме того, в доказательстве леммы 3 мы выяснили, что если на какой-то итерации добавление ребра $[u,v]$ сломало раскраску (она пропадала), то на любой последующей итерации добавление $[u,v]$ сломает раскраску. Из этого следует, что по окончании цикла мы имеем такой граф G, что:

- $C(G) = 1$ (т.е. $G \in 3 - COLOUR$)
- $\forall e \notin G \ : \ C(G + e) = 0$ (т.е. добавление любого нового ребра приводит к исчезновению раскраски)

По построению, мы имеем, что в G вершины u и v соединены ребром тогда и только тогда, когда $f(u,v) = 0$. Построим дополнение $\overline{G}$ к графу G. Матрица значений функции f будет матрицей смежности этого графа. Поэтому каждый класс эквивалентности отношения f будет соответствовать своей компоненте связности графа $\overline{G}$.

Пусть классов 4 или больше. Граф G можно окрасить в 3 цвета, значит, найдутся две компоненты связности одного цвета (по принципу Дирихле). Между ними в $\overline{G}$ нет ребер, так как это компоненты связности, значит, в G между ними есть ребра. Но тогда получится, что какие-то две вершины одного цвета соединены ребром. Противоречие. $\square$

Итак, мы доказали, что f действительно задает какую-то раскраску вершин графа G или G_0. Это следует из лемм 2, 3, 4. Теперь покажем, что f задает корректную раскраску графа G_0. На самом деле в лемме 4 мы это почти доказали, но вынесем это отдельно.

Лемма 5. *Получаемое в результате работы алгоритма отношение f задает корректную раскраску графа G_0 в 3 цвета (или меньше).*

Доказательство:
Из лемм 2,3,4 следует, что f есть раскраска в 3 цвета или меньше. По построению в алгоритме:

$$[u,v] \in G \qquad \Longleftrightarrow \qquad f(u,v) = 0,$$

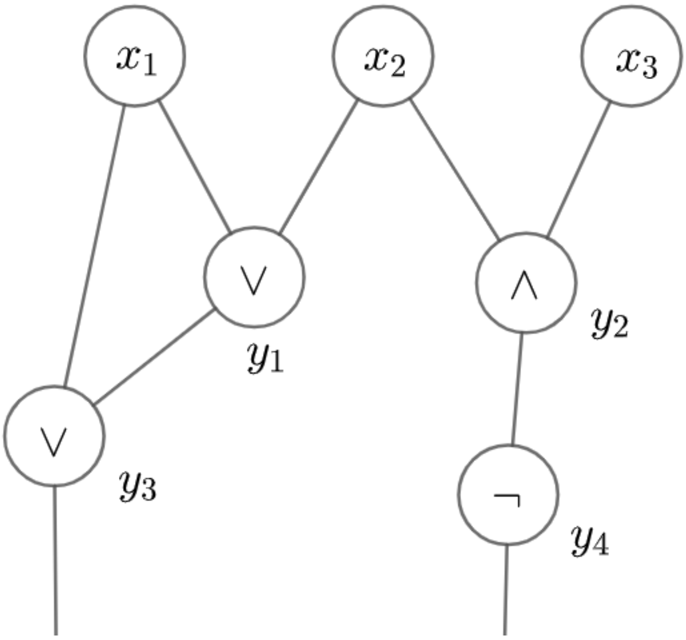
значит, раскраска G корректна. Но G_0 — подграф G, тогда, по лемме 1, раскраска G_0 тоже корректна. $\square$

Задача 7

Постройте сводимость языка CIRCUIT-SAT к 3-CNF.

Решение:

Будем считать, что в схеме S в вершинах стоят либо логические переменные $x_1, \dots, x_n$, либо операции $\vee, \wedge, \neg$. Пусть вершин с операциями всего m.



Преобразование f булевой схемы S в 3-КНФ ϕ:

1. Обозначим вершины с операциями новыми переменными $y_1, \dots, y_m$. Найдем значения этих переменных в зависимости от входов вершины:

$$\begin{cases} y_1 = x_1 \vee x_2 \\ y_2 = x_2 \wedge x_3 \\ y_3 = x_1 \vee y_1 \\ y_4 = \neg y_2 \\ \dots \end{cases}$$

   Замечание: вводим обозначения так, что y_k зависит только от y_s, где $s < k$, или от x_i. Это можно сделать, так как булева схема, будучи ориентированным графом, не содержит циклов. Легко придумать полиномиальный алгоритм введения таких обозначений (например, сделать обход в ширину, начиная с выхода схемы, а потом вводить обозначения вершин в обратном порядке).

2. Составим конъюнкцию m таких эквивалентностей, следующих из системы выше:

$$\phi_0 = \Big(y_1 \equiv (x_1 \vee x_2)\Big) \wedge \Big(y_2 \equiv (x_2 \wedge x_3)\Big) \wedge \dots \wedge \Big(y_m \equiv \dots\Big) \wedge y_m,$$

   где y_m — выход схемы.

3. Представим каждый член этой конъюнкции в виде полной КНФ, пользуясь соответствующим алгоритмом, и полученная большая 3-КНФ будет формулой ϕ.

Видно, что первый и второй шаги занимают полиномиальное время. Заметим, что на третьем шаге нам нужно преобразовывать в полную КНФ m формул константной длины (они содержат не более 3 литералов). Поэтому на каждую формулу уйдет не более, чем константное время. Отсюда следует, что третий шаг преобразования тоже требует полиномиального времени.

Также учтем, что полная КНФ эквивалентна самой формуле, поэтому формулы ϕ_0 и ϕ эквивалентны. Тогда достаточно доказать, что булева схема S выполнима тогда и только тогда, когда выполнима формула ϕ_0.

Пусть схема S выполнима. Тогда существует выполняющий набор $\alpha = (\alpha_1, \ldots, \alpha_n)$, при котором выход равен 1. С помощью системы из шага 1 алгоритма последовательно вычислим $y_1 = \beta_1, \ldots, y_m = \beta_m$. Схема выполнима, значит, $\beta_m = 1$. Ясно, что каждый другой член конъюнкции при наборе $\gamma = (\alpha_1, \ldots, \alpha_n, \beta_1, \ldots, \beta_m)$ тоже будет равен 1. Поэтому формула ϕ_0 выполнима при наборе γ.

Пусть формула ϕ_0 выполнима. Тогда существует выполняющий набор $\gamma = (\alpha_1, \ldots, \alpha_n, \beta_1, \ldots, \beta_m)$, $\beta_m = 1$. Формула выполнима, поэтому каждый член конъюнкции равен 1, т.е. верна система:

$$\begin{cases} \beta_1 = \alpha_1 \vee \alpha_2 \\ \beta_2 = \alpha_2 \wedge \alpha_3 \\ \beta_3 = \alpha_1 \vee \beta_1 \\ \ldots \\ \beta_m = \ldots \\ \beta_m = 1 \end{cases}$$

Пусть $\alpha = (\alpha_1, \ldots, \alpha_n)$ — вход булевой схемы S. Тогда последовательное вычисление значений в m вершинах схемы соответствует первым m уравнениям системы. $\beta_m = 1$ — значение на выходе, значит, схема S выполнима набором α.

Задача 10 (ДЗ 2)

Пусть $A \in \mathcal{NP} - complete$. Пусть машина имеет дополнительную функцию (оракул) за 1 такт получать ответ, лежит ли слово x в языке A. Тогда существует полиномиальный алгоритм, решающий задачу поиска для A. Докажите это утверждение.

Решение:

Мы знаем, что $3 - COLOUR \in \mathcal{NP} - complete,\ A \in \mathcal{NP} \implies$

$$A \leq_p 3 - COLOUR, \qquad f \text{ — сводящая функция, вычисляемая за полиномиальное время}$$

Нам дано, что $A \in \mathcal{NP} - complete,\ 3 - COLOUR \in \mathcal{NP} \implies$

$$3 - COLOUR \leq_p A, \qquad g \text{ — сводящая функция, вычисляемая за полиномиальное время}$$

Пусть $a(x)$ — оракул для языка A, вычисляемый за 1 такт.

Пусть $V(G, s)$ — верификатор для $3 - COLOUR$, где G — граф, а s — его раскраска (сертификат).

Построим оракул $C(G)$ для языка $3 - COLOUR$, вычисляемый за полиномиальное время.

$$G \in 3 - COLOUR \iff g(G) \in A \iff a\big(g(G)\big) = 1$$

Тогда $C(G) := a\big(g(G)\big)$ — полиномиальный оракул для $3 - COLOUR$.

В задаче 6 мы построили полиномиальный алгоритм $Q(G)$, который, используя полиномиальный оракул $C(G)$, находит сертификат-раскраску. Построим алгоритм $W(x)$, который будет искать сертификат для $x \in A$ — раскраску для графа $f(x)$:

$$W(x) := Q\big(f(x)\big)$$

Этот алгоритм работает за полиномиальное время. Покажем, что то, что он находит, действительно является сертификатом для $x \in A$. Построим верификатор $R(x, s)$:

$$R(x, s) := V(f(x), W(x)) = V\Big(f(x),\ Q\big(f(x)\big)\Big)$$

Ясно, что верификатор тоже работает за полиномиальное время.