

Первый курс, осенний семестр 2020/21  
Практика по алгоритмам #3

Первые сложности

29 января

Собрано 29 января 2021 г. в 09:54

---

Содержание

<b>1. Первые сложности</b>	<b>1</b>
<b>2. Разбор задач практики</b>	<b>3</b>
<b>3. Домашнее задание</b>	<b>7</b>
3.1. Обязательная часть . . . . .	7
3.2. Дополнительная часть . . . . .	8

# Первые сложности

## 1. Лежит ли задача в NP? А в coNP или P?

- SUBSET-SUM (KNAPSACK без стоимостей).
- KNAPSACK со стоимостями.
- GI – проверка изоморфизма двух графов.
- Проверка *отсутствия* пути  $a \rightsquigarrow b$  в графе.
- FACTOR <sub>$d$</sub>  – наличие делителя не более  $d$ .
- Поиск кратчайшей эквивалентной записи булевой формулы
  - Формула задана логическим выражением с OR, AND, NOT.
  - Формула задана таблицей истинности.

## 2. Предъявите для задачи decision-версию и сведите к ней search-версию

- MAX-SAT (максимизировать число выполненных кловов), найти сам набор переменных.
- HAMPATH
- 3-COLORING

## 3. Перебор

$NP \subseteq EXP$

## 4. Постройте сведение (полиномиальное или по Куку)

- HAMPATH  $\leftrightarrow$  HAMCYCLE
- 3-COLORING  $\rightarrow$  3-SAT
- 3-SAT  $\leftrightarrow$  NAE-3-SAT  
 NAE = not all equal значит, что в каждом клове хотя бы 1 ноль и хотя бы 1 единица.
- NAE-3-SAT  $\rightarrow$  MAX-CUT (дан взвешенный граф, разделить множество вершин на две непустых доли, максимизировать суммарный вес рёбер между долями).
- SAT-MAX-ONE (максимизировать число единиц)  $\rightarrow$  CIRCUIT-SAT

## 5. Доказать NP-полноту

- $(a, b)$ -CSP (constraint satisfaction problem) – есть  $n$  переменных, у каждой  $a$  возможных значений, есть  $m$  ограничений (запрещающих условий), каждое вида «такие то  $b$  переменных не могут одновременно принимать такие-то значения». Например, 3-SAT – частный случай  $(2, 3)$ -CSP.
- SAT (через уже доказанный CIRCUIT-SAT)
- MAX-3-SAT (выполнить, возможно, не все, но максимальное число кловов)
- VERTEX-COVER (выбрать минимальное число вершин, чтобы задеть все ребра)

## ♥ 6. coNEXP

Зная определения EXP, NP, coNP, определите класс coNEXP.

## 7. 3-SAT

Решить 3-SAT за  $\mathcal{O}^*(1.5^m)$ .

8. (\*) Решить 3-SAT за  $\mathcal{O}^*((2 - \varepsilon)^n)$ .

9. (\*) **Полиномиальные нижние оценки!**

OV = Orthogonal Vectors: даны вектора  $a_1, \dots, a_n, b_1, \dots, b_n$ , есть ли пара  $\langle a_i, b_j \rangle = 0$ ?  
Покажите SETH  $\Rightarrow$  OV нельзя решить за  $\mathcal{O}(n^{2-\varepsilon})$ .

† 10. (\*) HAM-CYCLE  $\in$  NPC

11. (\*) 3-SAT  $\rightarrow$  3-COLORING

12. (\*) 3-COLORING  $\rightarrow$  EXACT-SET-COVER

EXACT-SET-COVER: даны  $U$  и  $\{B_i \subseteq U\}$ , есть ли такое  $I: \cup_{i \in I} B_i = U \wedge \forall i, j B_i \cap B_j = \emptyset$ .

13. (\*) DOMINATING-SET  $\in$  NPC

DOMINATING-SET: найти  $\min A \subseteq V: A \cup N(A) = V$ .

14. (\*) STEINER-TREE  $\in$  NPC

STEINER-TREE: даны взвешенный граф и  $S \subseteq V$ , найти поддерево  $\min$  веса, содержащее все вершины  $S$ .

# Разбор задач практики

## 1. Лежит ли задача в NP? А в coNP или P?

- a) SUBSET-SUM. Подсказка для NP: набор предметов.
- b) KNAPSACK со стоимостями. Decision-версия: можно ли набрать цену больше  $C$ .  
Подсказка для NP: набор предметов.  
Решать за полином не умеют (динамика за  $\mathcal{O}(nS)$  работает за экспоненту от длины  $S$ ).
- c) GI. Подсказка для NP: перестановка, переводящая вершины первого графа в вершины второго. Умеют решать за  $2^{\text{poly}(\log n)}$ . Полагают, что не P и не NPC.
- d) Отсутствие пути в P (dfs)  $\Rightarrow$  в NP и в coNP (задачи из P решаются с пустой подсказкой).
- e) FACTOR<sub>d</sub>. Подсказка для NP и coNP: разложение на простые.  
Чтобы проверить подсказку для coNP, нужно уметь проверять на простоту за полином от длины числа. Это умеют за  $n^6$  ( $n = \log N$  – длина числа).  
FACTOR умеют за  $2^{Cn^{1/3}}$ . Полагают, что не P и не NPC.
- f) Кратчайшая запись булевой формулы.  
Короткая запись – не подсказка: непонятно, как проверить ее эквивалентность исходной формуле. Не знают, лежит ли в NP.  
Если формула задана таблицей истинности (длина входа  $2^n$ ), то проверим эквивалентность этой таблицы  $T$  и короткой записи  $\varphi$  перебором всех подстановок переменных. Время  $2^n \cdot |\varphi| = \mathcal{O}(\text{poly}(2^n)) = \mathcal{O}(|T|) \Rightarrow$  в NP.  
Если длина формулы  $\varphi$  больше  $2^n n$ , то она сразу не кратчайшая, ДНФ короче.

## 2. Decision $\rightarrow$ search

- a) MAX-SAT. Бинарным поиском ищем, сколько клозов можно удовлетворить, нашли  $k$ .  
Берем переменную  $x_1$ , подставляем ее равной 1 в формулу  $\varphi$ , получили  $\varphi'$ .  
Если верен MAX-SAT от  $\langle \varphi', k \rangle$ , то  $x_1 = 1$ , иначе  $x_1 = 0$ .  
Подставляем  $x_1$  и ищем остальные переменные.
- b) HAMPATH. По очереди попробуем каждое ребро удалить из графа.  
Если после удаления гамильтонов путь всё ещё есть, отлично, удалим ребро навсегда.  
*Альтернативное решение.* Будем строить путь от начала до конца: сперва переберём начало, затем будем перебирать продолжение. Пусть текущий кусок пути заканчивается на  $v$ . Удалим часть пути до  $v$ , а к  $v$  добавим фиктивный лист  $w$ . Если в новом графе есть гамильтонов путь, в исходном его можно было продолжить из  $v$ .
- c) 3-COLORING. Пока в графе больше трёх вершин, какие-то две должны быть покрашены в один цвет. Переберём, какие. Чтобы проверить, что их можно покрасить в один цвет, стянем их в одну вершину.

## 3. NP $\subseteq$ EXP: переберем подсказку.

## 4. Постройте сведение

- a) HAMPATH  $\leftrightarrow$  HAMCYCLE.  
HAMPATH  $\leq_p$  HAMCYCLE: добавим вершину  $a$ , соединим ее со всеми.  
Был путь в старом графе  $\Rightarrow$  цикл в новом графе: (путь в старом  $v - \dots - u$ )  $+(u - a - v)$ .

Есть цикл в новом графе  $\Rightarrow$  выкинем  $a$ , получим путь в старом графе.

Работает и для ор, и для неор. В ор надо соединять  $a$  со всеми в обе стороны.

НАМСУСЛЕ  $\leq_p$  НАМПАТН: раздвоим вершину 1 на две – in и out, из копии out только исходящие ребра, в in только входящие. Путь в новом графе обязан начаться в out (у нее нет входящих) и кончиться в in (нет исходящих).

Цикл в старом графе = путь в новом со склеенными in и out.

Для неорграфа надо проводить из in и out все ребра, но подвесить к in и out по листу.

#### b) 3-COLORING $\rightarrow$ 3-SAT.

Переменные  $x_{vc}$  – покрашена ли  $v$  в  $c$ .

Ребро  $(u, v)$  для каждого цвета  $c$  дает кюз  $(\neg x_{uc} \vee \neg x_{vc})$ . Получили корректную покраску.

Для каждой  $v$  добавим кюз  $(x_{v1} \vee x_{v2} \vee x_{v3})$ . Теперь  $v$  покрашена хотя бы в один цвет.

Покраска и удовлетворяющий набор переменных легко получаются друг из друга.

#### с) 3-SAT $\leftrightarrow$ NAE-3-SAT.

NAE-3-SAT  $\rightarrow$  3-SAT: запишем для каждого кюза  $(l_1 \vee l_2 \vee l_3)$  парный  $(\neg l_1 \vee \neg l_2 \vee \neg l_3)$ . Решим SAT из  $2m$  кюзов.

3-SAT  $\rightarrow$  NAE-4-SAT: добавим в каждый кюз переменную  $w$  (одну и ту же).

Было решение 3-SAT  $\Rightarrow$  берем его и делаем  $w = 0$ .

Есть решение  $(y_1, y_2, \dots, y_n, w)$  для NAE-4-SAT  $\Rightarrow$  ответ для 3-SAT это  $x_i = y_i \wedge w$ .

NAE-4-SAT  $\rightarrow$  NAE-3-SAT:  $i$ -й кюз из 4 литералов  $(l_1 \vee l_2 \vee l_3 \vee l_4)$  переводим в кюзы  $(l_1 \vee l_2 \vee w_i)$ ,  $(\neg w_i \vee l_3 \vee l_4)$ ,  $w_i$  – новая переменная (так же, как в 4-SAT  $\rightarrow$  3-SAT).

Чтобы перевести решение NAE-4-SAT в решение NAE-3-SAT и обратно, разбор 5 случаев:  $l_1 = l_2 = 0/1$ ,  $l_3 = l_4 = 0/1$ , иначе.

#### d) NAE-3-SAT $\rightarrow$ MAX-CUT

Для каждой переменной  $x_i$  заводим две вершины:  $x_i, \bar{x}_i$ , соединяем их ребром.

Каждый кюз преобразуем в три ребра:  $\Delta$  на литералах.

Получили граф из  $2n$  вершин и  $3m + n$  ребер.

Проверим существование разреза размера хотя бы  $n + 2m$ , больше точно не бывает.

Если такой есть, все  $x_i$  и  $\bar{x}_i$  лежат по разные стороны разреза, и в каждом треугольнике ровно два ребра в разрезе (для этого и нужно NAE).

#### e) SAT-MAX-ONE $\rightarrow$ CIRCUIT-SAT

Преобразуем сперва SAT к CIRCUIT-SAT, обозначим выходной гейт  $v_{\text{out}}$ .

Создадим гейты  $f_{i,j}$  – среди первых  $i$  переменных хотя бы  $j$  единиц.

$f_{i,j} = f_{i-1,j} \vee (f_{i-1,j-1} \wedge x_i)$ . Ответ в  $v_{\text{out}} \wedge f_{n,k}$ .

*Альтернативное решение.*

А можно честно научиться прибавлять  $x$  в двоичной системе счисления.

```

1 carry0 = x
2 for ai // перебираем цифры суммы с младших разрядов
3   carryi+1 = carryi and ai // гейты для переносов
4   bi = carryi xor ai // b - новая сумма, гейты для новой суммы

```

5. Чтобы доказать NP-полноту, нужно свести какую-либо NP-полную задачу к данной.

a)  $(a, b)$ -CSP. 3-SAT – частный случай CSP, свелось.

b) SAT. Сведём к нему CIRCUIT-SAT: каждому узлу соответствует формула  $v_i = f_i(v_j, v_k)$ . Это булева формула от трех переменных, запишем ее в КНФ. Соединим все КНФ вместе.

c) MAX-3-SAT. Сводим к ней 3-SAT:  $\varphi \rightarrow \langle \varphi, m \rangle$ , выполнено хотя бы  $m$  кловов, то есть все.

d) VERTEX-COVER. Сведём к нему INDEPENDENT-SET:  $\langle G, k \rangle \rightarrow \langle G, n - k \rangle$ .

Множество – вершинное покрытие  $\Leftrightarrow$  его дополнение – независимое множество.

6.  $\text{coNEXP} = \{L: \exists M \in \text{EXP-TIME } \forall x (x \notin L) \Leftrightarrow (\exists y M(x, y) = 1)\}$ .

Эквивалентно  $\{L: \exists M \in \text{EXP-TIME } \forall x (x \in L) \Leftrightarrow (\forall y M(x, y) = 0)\}$ .

7. Решить 3-SAT за  $\mathcal{O}^*(1.5^m)$ .

Так же, как в прошлом семестре решали 3-LIST-COLORING.

Выкинем из каждого клова случайный литерал  $\Rightarrow$  с вероятностью  $\geq (\frac{2}{3})^m$  получим решаемый 2-SAT. Повторим  $1.5^m$  раз, получим вероятность ошибки  $e^{-1}$ .

8. (\*) Решить 3-SAT за  $\mathcal{O}^*((2 - \varepsilon)^n)$ .

Возьмём любой клов с тремя литералами, есть 8 вариантов присвоить переменным значения, но только 7 из них обратят клов в истину. Получили  $T(n) = 7T(n - 3) = 7^{n/3} \approx 1.91^n$ .

Также можно решать и  $k$ -SAT за  $\mathcal{O}((2^k - 1)^{n/k})$ .

9. (\*) **Полиномиальные нижние оценки!**

[stanford.edu]

SAT  $\rightarrow$  OV. Пусть дана формула с кловами  $C_1, \dots, C_m$ .

Рассмотрим переменные  $x_1, \dots, x_{n/2}$ . Рассмотрим все  $2^{n/2}$  подстановок этих переменных.

Подстановке  $s$  сопоставим вектор  $a_s = \langle a_{s1}, \dots, a_{sm} \rangle$ :  $a_{si} = \neg(C_i \text{ выполнен подстановкой } s)$ .

Аналогично строим набор из  $2^{n/2}$  векторов  $b_t$  по второй половине переменных.

$\langle a_s, b_t \rangle = 0 \Leftrightarrow \forall i (a_{si} = 0 \vee b_{ti} = 0) \Leftrightarrow$  подстановкой  $st$  удовлетворены все кловы.

Решаем OV за  $n^{2-\varepsilon} \Rightarrow$  решаем  $\forall k$ -SAT за  $\mathcal{O}((2^{n/2})^{2-\varepsilon}) = \mathcal{O}(2^{(1-\frac{\varepsilon}{2})n})$ . Противоречие с SETH.

*Note: длина векторов  $m$  должна быть мала относительно их количества:*

$m = \text{poly}(n) = \text{poly}(\log(2^{n/2}))$  подойдёт.

10. (\*) HAMCYCLE  $\in$  NPC. Сведём 3-SAT. **Пара картинок.**

11. (\*) 3-SAT  $\rightarrow$  3-COLORING

Заведём вершины  $T, F, N$  (True, False, Neutral) и попарно соединим рёбрами.

Теперь они должны быть разных цветов, которые так и назовем:  $T, F, N$ .

$\forall i$  заведём вершины  $x_i, \neg x_i$  и рёбра  $(x_i, \neg x_i), (x_i, N), (\neg x_i, N)$ .

Теперь одна из  $x_i, \neg x_i$  должна быть  $T$ , другая  $F$ .

Научимся для вершин  $a, b$  строить некое подобие OR.

Добавим вершины  $u, v, z$ , рёбра  $(a, u), (b, v)$  и треугольник  $(u, v, z)$ .

Если  $a = b = F$ , то  $z$  тоже обязательно красить в  $F$ . Иначе **существует** корректная покраска  $u$  и  $v$ , что  $z \neq F$ .

Если повторить эту конструкцию два раза  $((a \vee b) \vee c)$ , то при  $a = b = c = F$  выход обязательно  $F$ , иначе **существует** корректная покраска, в которой выход –  $T$ .

Осталось соединить выход с  $N$  и  $F$ . Добавляем такую конструкцию для каждого кноза.

Подробнее и с картинками в [pdf](#).

## 12. (\*) 3-COLORING $\rightarrow$ EXACT-SET-COVER

Универсум: все вершины, все ребра и все тройки  $\langle v, e, c \rangle$ , где  $v$  – конец  $e$ ,  $c$  – один из цветов (итого  $n + m + 6m$  элементов). Множества:

а)  $\forall$  вершины  $v$ , цвета  $c$ :  $\{v\} \cup \{\langle v, e, c \rangle \mid e = (v, u)\}$ .

Если взято, то красим  $v$  в цвет  $c$ . Каждая вершина покрасится, причем в один цвет.

б)  $\forall$  ребра  $e = (v, u)$ , пары цветов  $c_1 \neq c_2$ :  $\{e\} \cup \{\langle v, e, c \rangle \mid c \neq c_1\} \cup \{\langle u, e, c \rangle \mid c \neq c_2\}$ .

Раз каждое ребро чем-то покрыто, то оно «забирает с собой» все цвета у своих концов, кроме двух неравных.

## 13. (\*) DOMINATING-SET $\in$ NPC: VERTEX-COVER $\rightarrow$ DOMINATING-SET.

Из графа  $G = (V, E)$  делаем новый  $G' = (V', E')$ .

$V' = V \cup E$ .  $E' = \{(v, u) \mid v, u \in V\} \cup \{(v, e) \mid e = (v, u) \in E\}$ .

Чтобы задоминировать все вершины, которые соответствуют старым ребрам, нужно как раз набрать VERTEX-COVER. Полный граф на старых вершинах нужен, чтобы они сами собой задоминировались.

## 14. (\*) STEINER-TREE $\in$ NPC: SET-COVER $\rightarrow$ STEINER-TREE.

Вершины графа – элементы и множества.

Ребра веса 0 между множествами и лежащими в них элементами.

Фиктивная вершина  $v_0$ , соединенная ребрами веса 1 со всеми множествами.

$S$  – множество элементов. Вес  $\min$  дерева = (количеству множеств в  $\min$  SET-COVER – 1).

# Домашнее задание

## 3.1. Обязательная часть

Мы уже доказали NP-полноту задач BOUNDED HALTING, CIRCUIT-SAT, SAT, MAX-SAT, INDEPENDENT-SET, CLIQUE, VERTEX COVER, MAX-CUT, 3-COLORING. Примем на веру NP-полноту HAMPATH. При решении можно опираться на их NP-полноту.

### 1. (1) Лежит ли в NP следующая задача?

«Последовательность  $a$  – не подпоследовательность последовательности  $b$ ». Ответ обоснуйте.

*Вы же были на практике, да? Простая задача. Разбор поможет.*

### 2. (2) MAX-SAT $\rightarrow$ CIRCUIT-SAT.

MAX-SAT – выполнить максимальное количество клозов в SAT.

*Вы же были на практике, да? Простая задача. Разбор поможет.*

### 3. (2) MAX-CUT.

Через decision версию задачи решите search версию задачи.

Нужно найти величину разреза и сам разрез.

*Вы же были на практике, да? Простая задача. Разбор поможет.*

### 4. (2) EXACT-SET-COVER $\rightarrow$ SUBSET-SUM.

EXACT-SET-COVER: даны  $U$  и  $\{B_i \subseteq U\}$ , есть ли такое  $I: \cup_{i \in I} B_i = U \wedge \forall i, j B_i \cap B_j = \emptyset$ .  
SUBSET-SUM: набрать сумму ровно  $S$ .

*Тут и дальше придётся думать.*

### 5. (2) Decision версия SET-COVER $\in$ NPC.

SET-COVER: выбрать минимальное число множеств, объединение которых равно  $A$ .

Decision версия: ответ  $\leq x$ .

### 6. (2) Decision версия HITTING-SET $\in$ NPC.

HITTING-SET: даны  $n$  подмножеств универсума  $U$ , выбрать минимальное число элементов  $U$ , чтобы в каждом из  $n$  множеств был выбран хотя бы один элемент. Decision версия: ответ  $\leq x$ .

*Бесплатные баллы!*

### 7. (2) Decision TSP $\in$ NPC (задача коммивояжера).

TSP – выбрать в *полном* взвешенном орграфе гамильтонов цикл наименьшей длины.

Приведите decision версию, докажите её NP-полноту.

### 8. (1) Докажите: ETH $\Rightarrow$ P $\neq$ NP.



## 3.2. Дополнительная часть

### 1. (2) SUBSET-SUM $\rightarrow$ PARTITION $\rightarrow$ JOB-SCHEDULING

PARTITION: разделить множество предметов на два, равных по суммарному весу.

JOB-SCHEDULING: даны  $n$  заказов, у каждого дано время выполнения  $t_i$ . Есть  $k$  одинаковых рабочих, распределить заказы по рабочим, минимизировав время завершения всех работ.

### 2. (2) 0-1-ILP $\in$ NPC.

ILP – целочисленное линейное программирование. Есть переменные  $x_1, \dots, x_n$ .

Даны неравенства  $\langle a_i, x \rangle \leq b_i$  (скалярные произведения) и вектор  $c$ .

Найти  $x_i \in \{0, 1\}$ , максимизирующие  $\langle c, x \rangle$ , удовлетворяющие всем неравенствам.

### 3. (3) PRIME $\in$ NP. Не используя то, что PRIME $\in$ P =)

### 4. (3) MAX-2-SAT $\in$ NPC

### 5. (3+2) Полиномиальные нижние оценки!

а) (3) Покажите SETH  $\Rightarrow$  нельзя найти доминирующее множество размера  $k$  за  $\mathcal{O}(n^{k-\varepsilon})$ .

б) (2) Покажите SETH  $\Rightarrow$  нельзя найти доминирующее множество размера  $k$  за  $\mathcal{O}(n^{k-1-\varepsilon})$  в графах с маленькой средней степенью ( $m = \mathcal{O}(n \text{polylog}(n))$ ).