SPb HSE, 1 курс, зима 2024/25
Практика по алгоритмам #16

Обыкновенная сложность
28 января 2025

Собрано 27 января 2025 г. в 21:58

Содержание

Обыкновенная сложность

1. **Поллард**

   Вспомнить и улучшить $\mathcal{O}(n^{1/4} \log n)$ до $\mathcal{O}(n^{1/4})$.

2. **Тривиальные вложения**

   a) Докажите $\mathrm{P} \subseteq \mathrm{NP} \cap \mathrm{coNP}$ (простейшая)
   b) $\mathrm{P} = \mathrm{NP} \Rightarrow \mathrm{P} = \mathrm{coNP}$ (разбор дз)

3. **Перебор**

   $\mathrm{NP} \subseteq \mathrm{EXP}$ (повтор практики)

4. **Иерархия по времени и будильники**

   Докажите, что нельзя за $\mathcal{O}(n)$ определить, остановится ли программа за n^2 шагов. $n = |x|$.

5. **Постройте сведение (полиномиальное или по Куку)**

   a) HAMPATH $\leftrightarrow$ HAMCYCLE (в прошлой практике было сведение в одну из сторон).
   b) 3-COLORING $\to$ 3-SAT
   c) SAT-MAX-ONE (максимизировать число единиц) $\to$ CIRCUIT-SAT

6. **Доказать NP-полноту**

   a) SAT (через уже доказанный CIRCUIT-SAT) (было на лекции, повторение)
   b) MAX-3-SAT (выполнить не обязательно все, но максимальное число клозов)
   c) VERTEX-COVER (разбор дз; выбрать min число вершин, чтобы задеть все ребра)
   d) ILP (разбор дз; целочисленное линейное программирование)

7. **Предъявите для задачи decision-версию и сведите к ней search-версию**

   a) MAX-SAT (максимизировать число выполненных клозов), найти сам набор переменных.
   b) HAMPATH
   c) 3-COLORING

♥ 8. **coNEXP**

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

9. **Решить** 3-SAT

   За $\mathcal{O}^*((2-\varepsilon)^n)$.

10. **(*) Полиномиальные нижние оценки!**

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

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

† 12. **(*)** HAM-CYCLE $\in$ NPc

Разбор задач практики

1. **Поллард**

```
b = a = random
while True:
    a = f(a)
    b = f(f(b))
    if gcd(a - b, n) != 1: // нашли делитель
```

Можно считать gcd не на каждом шаге, а сразу посчитать gcd от $\prod(a_i - b_i)$, взять в произведение $\geqslant \log n$ разностей.

2. **Тривиальные вложения**

a) P $\subseteq$ NP: берём чекер $M(x,y)$, который просто решает задачу
P $\subseteq$ coNP: берём чекер $M(x,y)$, который просто решает задачу

b) $\Rightarrow$ P $\subseteq$ NP $\cap$ coNP
$L \in$ coNP $\Rightarrow \overline{L} \in$ NP $=$ P $\Rightarrow \exists M(x)$, решающий $\overline{L}$ за полином, он же решит L.

Иначе: $\forall L \in NP\ \overline{L} \in coNP$ (NP – есть подсказка, что x «хороший», лежит в L; coNP – есть подсказка, что x «плохой», не лежит в L). $\forall L \in NP \exists A \in P$: $(A(x) = 1 \Leftrightarrow x \in L) \Rightarrow (A(x) = 0 \Leftrightarrow x \notin L)$. Простроили алгоритм $A'(x) = 1 - A(x)$, доказали, что $\overline{L} \in$ coNP.

3. NP $\subseteq$ EXP**:** по определению NP $\exists$ чекер M, переберем подсказку y, $\forall y$ запустим $M(x, y)$.

4. **Иерархия по времени**

Пусть есть решение $H(A,x)$, работающее за $\mathcal{O}(|x|)$ и возвращающее 1,
если программа A остановится на x за $|x|^2$ шагов. Рассмотрим F:

```
def F(A):
    if H(A,A) == 0: return // не остановилась за n^2 шагов
    поработай |A|^3
```

Пытаемся запустить $F(F)$, смотрим, что будет:
Если $H(F,F) = 1 \Rightarrow F(F)$ работает долго и $H(F,F)$ должно вернуть 0 ?!?
Если $H(F,F) = 0 \Rightarrow F(F)$ работает быстро и $H(F,F)$ должно вернуть 1 ?!?

5. **Постройте сведение**

a) HAMPATH $\leftrightarrow$ HAMCYCLE.

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

HAMCYCLE $\leqslant_p$ HAMPATH: 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 → 3-SAT.

Переменные x_{vc} – покрашена ли v в c.
Ребро (u, v) для каждого цвета c дает клоз $(\neg x_{uc} \vee \neg x_{vc})$. Получили корректную покраску.
Для каждой v добавим клоз $(x_{v1} \vee x_{v2} \vee x_{v3})$. Теперь v покрашена хотя бы в один цвет.
Покраска и удовлетворяющий набор переменных легко получаются друг из друга.

c) SAT-MAX-ONE → CIRCUIT-SAT

Преобразуем сперва SAT к CIRCUIT-SAT, обозначим выходной гейт $v_{\texttt{out}}$.
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 $v_{\texttt{out}} \wedge f_{n,k}$.

Альтернативное решение.
А можно честно научиться прибавлять x в двоичной системе счисления.

```
1 carry_0 = x
2 for a_i // перебираем цифры суммы с младших разрядов
3     carry_{i+1} = carry_i and a_i // гейты для переносов
4     b_i = carry_i xor a_i // b - новая сумма, гейты для новой суммы
```

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

a) 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.
b) MAX-3-SAT. Сводим к ней 3-SAT: $\varphi \to \langle \varphi, m \rangle$, выполнено хотя бы m клозов, то есть все.
c) VERTEX-COVER. Сведём к нему INDEPENDENT-SET: $\langle G, k \rangle \to \langle G, n - k \rangle$.
Множество – вершинное покрытие ⇔ его дополнение – независимое множество.
d) 0-1-ILP $\in$ NPC.
VERTEX-COVER → ILP. Каждой вершине v сопоставляем переменную x_v – взята вершина в покрытие или нет. Неравенства: для каждого ребра (v, u) пишем $x_v + x_u \geqslant 1$, то есть хотя бы один конец ребра взят. Минимизировать $\sum x_v$.

7. **Предъявите для задачи decision-версию и сведите к ней search-версию**

a) MAX-SAT. Бинпоиском ищем, сколько клозов можно удовлетворить, нашли k.
Берем переменную x_1, подставляем ее равной 1 в формулу φ, получили φ'.
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.

8. coNEXP $= \{L\colon \exists M \in \text{EXP-TIME}\ \forall x\ (x \notin L) \Leftrightarrow (\exists y\, M(x,y) = 1)\}$.
   Эквивалентно $\{L\colon \exists M \in \text{EXP-TIME}\ \forall x\ (x \in L) \Leftrightarrow (\forall y\, M(x,y) = 0)\}$.

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

   Возьмём любой клоз с тремя литералами, есть 8 вариантов присвоить переменным значения, но только 7 из них обратят клоз в истину. Получили $T(n) = 7T(n-3) = 7^{n/3} \approx 1.91^n$.
   Также можно решать и k-SAT за $\mathcal{O}((2^k-1)^{n/k})$.

10. (*) **Полиномиальные нижние оценки!**

    [stanford.edu]

    SAT $\to$ OV. Пусть дана формула с клозами $C_1, \ldots, C_m$.
    Рассмотрим переменные $x_1, \ldots, x_{n/2}$. Рассмотрим все $2^{n/2}$ подстановок этих переменных.
    Подстановке s сопоставим вектор $a_s = \langle a_{s1}, \ldots, a_{sm}\rangle$: $a_{si} = \neg(C_i$ 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 клозы.
    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.*

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

    Заведем вершины T, F, N (`True`, `False`, `Neutral`) и попарно соединим рёбрами.
    Теперь они должны быть разных цветов, которые так и назовем: T, F, N.
    $\forall i$ заведем 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 $a{=}b{=}F \Rightarrow z$ тоже обязательно красить в F. Иначе $\exists$ корректная покраска $u, v, z\colon z \neq F$.
    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 это T.
    Осталось соединить выход с N и F. Добавляем такую конструкцию для каждого клоза.
    Кстати, на самом деле мы сделали SAT $\to$ 3-COLORING.

    Подробнее и с картинками в pdf.

12. (*) HAMCYCLE $\in$ NPc. Сведём 3-SAT. Пара картинок.

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.

2. **(2) Сведение.** MAX-SAT $\to$ CIRCUIT-SAT

   MAX-SAT – выполнить максимальное количество клозов в SAT.
   Простая задача. Разбор практики поможет.

3. **(2) Полнота #1.** SET-COVER $\in$ NPc

   SET-COVER: есть универсум U из m элементов, дано n его подмножеств $A_1, \ldots, A_n \subseteq U$. Выбрать минимальное число множеств из $\{A_i\}$, чтобы их объединение было равно U.
   Decision-версия: правда ли, что ответ $\leqslant k$?
   Докажите, что decision-версия задачи SET-COVER лежит в NPc.

4. **(2) Полнота #2.** Decision TSP $\in$ NPc (задача коммивояжера).

   TSP – выбрать в *полном* взвешенном орграфе гамильтонов цикл наименьшего суммарного веса. Приведите decision версию, докажите её NP-полноту.

5. **(1) Объединение, пересечение и конкатенация в** NP

   Мы уже знаем, что:
   a) NP замкнуто относительно полиномиального сведения: $A \leqslant_p B$, $B \in \text{NP} \Rightarrow A \in \text{NP}$.
   b) NP вряд ли замкнуто относительно дополнения: мы предполагаем, что $\text{NP} \neq \text{coNP}$.

   Докажите, что:
   a) **(0.5)** NP замкнуто относительно объединения: $A, B \in \text{NP} \Rightarrow A \cup B \in \text{NP}$.
   b) **(0.5)** NP замкнуто относительно пересечения: $A, B \in \text{NP} \Rightarrow A \cap B \in \text{NP}$.
   c) **(+0.5)** (доп) NP замкнуто относительно конкатенации: $A, B \in \text{NP} \Rightarrow AB \in \text{NP}$. $AB = \{ab \mid a \in A, b \in B\}$, где ab — строка, полученная конкатенацией строки a и строки b.

3.2. Дополнительная часть

1. **(1)** coNP**-трудность** Докажите, что L – NP-трудная задача $\iff$ $\overline{L}$ – coNP-трудная задача ($\forall$ задача из coNP полиномиально сводится к ней).

2. **(1)** HITTING-SET $\in$ NPc

   HITTING-SET: есть универсум U из m элементов, дано n его подмножеств $A_1, \ldots, A_n \subseteq U$. Выбрать минимальное число элементов U, чтобы в каждом из n множеств был выбран хотя бы один элемент. Decision версия: правда ли, что ответ $\leqslant k$? Докажите, что decision-версия задачи HITTING-SET лежит в NPc.

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