Первый курс, весенний семестр 2023/24
Практика по алгоритмам #3

Первые сложности

1 февраля 2024

Собрано 13 февраля 2024 г. в 20:05

Содержание

Первые сложности

1. **Лежит ли задача в** NP**? А в** coNP **или** P**?**
   a) SUBSET-SUM (KNAPSACK без стоимостей).
   b) KNAPSACK со стоимостями.
   c) GI – проверка изоморфизма двух графов.
   d) Проверка *отсутствия* пути $a \rightsquigarrow b$ в графе.
   e) FACTOR_d – наличие делителя не более d.
   f) Поиск кратчайшей эквивалентной записи булевой формулы
      1. Формула задана логическим выражением с `OR`, `AND`, `NOT`.
      2. Формула задана таблицей истинности.

2. **Тривиальные вложения**
   a) Докажите $\text{P} \subseteq \text{NP} \cap \text{coNP}$.
   b) $\text{P} = \text{NP} \Rightarrow \text{P} = \text{coNP}$.

3. **Перебор**
   $\text{NP} \subseteq \text{EXP}$

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
   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)
   b) MAX-3-SAT (выполнить не обязательно все, но максимальное число клозов)
   c) VERTEX-COVER (выбрать минимальное число вершин, чтобы задеть все ребра)

♥ 7. **coNEXP**
   Зная определения EXP, NP, coNP, определите класс coNEXP.

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

† 9. (*) HAM-CYCLE $\in$ NPc

10. (*) 3-SAT $\to$ 3-COLORING

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$_d$. 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 φ перебором всех подстановок переменных. Время $2^n \cdot |\varphi| = \mathcal{O}(\text{poly}(2^n)) = \mathcal{O}(|T|) \Rightarrow$ в NP.
   Если длина формулы φ больше $2^n n$, то она сразу не кратчайшая, ДНФ короче.

2. **Тривиальные вложения**

   a) $\text{P} \subseteq \text{NP}$: берём чекер $M(x,y)$, который просто решает задачу
   $\text{P} \subseteq \text{coNP}$: берём чекер $M(x,y)$, который просто решает задачу
   $\Rightarrow \text{P} \subseteq \text{NP} \cap \text{coNP}$
   b) $L \in \text{coNP} \Rightarrow \overline{L} \in \text{NP} = \text{P} \Rightarrow \exists M(x)$, решающий $\overline{L}$ за полином, он же решит L.

3. $\text{NP} \subseteq \text{EXP}$**:** переберем подсказку.

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) == 1: return
  поработай |A|^3
```

   $H(F,F) = 1 \Rightarrow F$ работает долго и $H(F,F) = 0 \Rightarrow F$ работает быстро и $H(F,F) = 1$!?!

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 $\to$ 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 $\to$ CIRCUIT-SAT

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

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

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

6. 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: $\varphi \to \langle \varphi, m \rangle$, выполнено хотя бы m клозов, то есть все.
d) VERTEX-COVER. Сведём к нему INDEPENDENT-SET: $\langle G, k \rangle \to \langle G, n-k \rangle$.
Множество – вершинное покрытие $\Leftrightarrow$ его дополнение – независимое множество.

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

8. `(*)` 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 $\mathcal{O}((2^k-1)^{n/k})$.

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

10. `(*)` 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 $a = b = F$, 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 клоза.

    Подробнее и с картинками в pdf.

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 $A_1, \dots, A_n \subseteq U$.
   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.** HITTING-SET $\in$ NPc

   HITTING-SET: есть универсум U из m элементов, дано n его подмножеств $A_1, \dots, A_n \subseteq U$.
   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.

5. **(2) Полнота #3.** Decision TSP $\in$ NPc (задача коммивояжера).

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

6. **(1)** coNP**-трудность**

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

7. **(1.5) Объединение, пересечение и конкатенация в** NP

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

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

3.2. Дополнительная часть

1. **(3) Сложная подсказка.** PRIME $\in$ NP. Не используя то, что PRIME $\in$ P =)
2. **(2) Полнота #4.** 0-1-ILP $\in$ NPc.

   ILP – целочисленное линейное программирование. Есть переменные $x_1, \ldots, x_n$.
   Даны неравенства $\langle a_i, x\rangle \leqslant b_i$ (скалярные произведения) и вектор c.
   Найти $x_i \in \{0,1\}$, максимизирующие $\langle c, x\rangle$, удовлетворяющие всем неравенствам.
   Здесь a_i, x, b, c — вектора из нулей и единиц.