Первый курс, весенний семестр 2023/24
Практика по алгоритмам #4

Вероятностные сложности

8 февраля 2024

Собрано 13 февраля 2024 г. в 20:05

Содержание

1. Вероятностные сложности **1**

2. Разбор задач практики **3**

3. Домашнее задание **6**

3.1. Обязательная часть 6

3.2. Дополнительная часть 7

Вероятностные сложности

1. **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

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

   a) 3-SAT $\leftrightarrow$ NAE-3-SAT
   NAE = not all equal значит, что в каждом клозе хотя бы 1 ноль и хотя бы 1 единица.
   b) NAE-3-SAT $\to$ MAX-CUT (дан взвешенный граф, разделить множество вершин на две непустых доли, максимизировать суммарный вес рёбер между долями).

3. **3-List-Coloring**

   3-List-Coloring: есть граф и k цветов, но для каждой вершины дан список из трех цветов, в которые ее можно красить. Найти правильную покраску или сказать, что такой нет. $\mathcal{O}(1.5^V E)$.

4. **Понижение вероятности**

   Алгоритм работает за $\mathcal{O}(n^2 \log n)$, вероятность успеха $1/\log n$.
   За какое время можно добиться вероятности успеха $1 - 1/n$?

5. **Числа в окрестности**

   Дан массив и число ε. Известно, что $\frac{2n}{3}$ элементов массива лежат в $[x, x + \varepsilon]$ для некоторого **неизвестного** x. За линейное время выбрать $\frac{n}{3}$ элементов из $[y, y + \varepsilon]$ для некоторого y.

6. **Приближение** MAX-SAT

   Для MAX-SAT построить полиномиальный вероятностный алгоритм, ищущий набор переменных, обращающий хотя бы $\frac{1}{2}\texttt{OPT}$ клозов в истину.

7. **Random в задачах оптимизации**

   Пусть у нас есть алгоритм, который ищет VERTEX-COVER.
   На любом графе матожидание размера покрытия, которое выдаст алгоритм, равно $C \cdot \texttt{OPT}$.
   Как получить покрытие, которое с высокой вероятностью будет размера не более $C(1+\varepsilon)\texttt{OPT}$?

8. **Первообразный корень**

   Первообразный корень по модулю p – такое x: $\langle 1, x, x^2, \ldots, x^{p-2} \rangle$ различны. Дано p, найти x.

9. (*) **Проверка ДЗ**

   Дано ДЗ из $n{=}2k$ задач. Пусть студент в каждой из задач делает ошибку с вероятностью $\frac{1}{3}$. Ошибки во всех задача независимы.

   Алгоритм #1 проверки дз: проверить первую половину присланных решений. Пусть обнаружено 0 ошибок, какова вероятность наличия ошибок/матожидание количества ошибок?

   Алгоритм #2 проверки дз: проверить *случайную* половину присланных решений. Пусть обнаружено 0 ошибок, какова вероятность наличия ошибок/матожидание количества ошибок?

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-COLORING $\to$ EXACT-SET-COVER

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

12. (*) DOMINATING-SET $\in$ NPc

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

13. (*) 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. **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.

2. **Постройте сведение**

   a) 3-SAT $\leftrightarrow$ NAE-3-SAT.

   NAE-3-SAT $\to$ 3-SAT: запишем для каждого клоза $(l_1 \vee l_2 \vee l_3)$ парный $(\neg l_1 \vee \neg l_2 \vee \neg l_3)$.
   Решим SAT из $2m$ клозов.

   3-SAT $\to$ NAE-4-SAT: добавим в каждый клоз переменную w (одну и ту же).
   Было решение 3-SAT $\Rightarrow$ берем его и делаем $w = 0$.
   Есть решение $(y_1, y_2, \ldots, y_n, w)$ для NAE-4-SAT $\Rightarrow$ ответ для 3-SAT это $x_i = y_i \hat{} w$.

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

   b) NAE-3-SAT $\to$ MAX-CUT

   Дана NAE-3-SAT формула, нам нужно построить граф.

   Для каждой переменной x_i заводим две вершины: x_i, $\overline{x_i}$, соединяем их ребром.
   Каждый клоз преобразуем в три ребра: $\triangle$ на литералах.
   Получили граф из $2n$ вершин и $3m + n$ рёбер.

   Проверим существование разреза размера хотя бы $n + 2m$, больше точно не бывает.
   Если такой есть, все x_i и $\overline{x_i}$ лежат по разные стороны разреза, и в каждом треугольнике ровно два ребра в разрезе (для этого и нужно NAE).

3. **3-List-Coloring**

   Если запретить случайный цвет в каждой вершине, то получится `2-List-Coloring`, который сводится к `2-SAT` и решается за $\mathcal{O}(E)$.

Если есть правильная покраска, то у каждой вершины мы запретили ее цвет с вероятностью $\leqslant \frac{1}{3} \Rightarrow$ с вероятностью $\geqslant (\frac{2}{3})^V = p$ у всех вершин не запрещён нужный цвет.

Получили решение за $\mathcal{O}(E)$, которое работает с вероятностью не менее p. Если повторить его $\frac{1}{p}$ раз, вероятность неудачи будет не более $(1-p)^{1/p} \approx e^{-1}$. Если повторить его $20\frac{1}{p} = \mathcal{O}(1.5^V)$ раз, вероятность неудачи будет не более $e^{-20} \approx 2 \cdot 10^{-9}$.

4. **Понижение вероятности**

После $\log n$ повторов вероятность ошибки $(1 - \frac{1}{\log n})^{\log n} \leqslant e^{-1}$.
Всё это вместе $\ln n$ раз, тогда $\frac{1}{n}$. Итого $\log n \cdot \ln n$ повторов, время работы $\mathcal{O}(n^2 \log^3 n)$.

5. **Числа в окрестности**

Простой способ.
Ткнем в случайное число a_i, выведем в ответ все числа из $[a_i..a_i + \varepsilon]$.
Обозначим за B минимальные $\frac{n}{3}$ чисел из $[x..x+\varepsilon]$. Если $a_i \in B$, то на $[a_i..a_i+\varepsilon]$ лежит хотя бы $\frac{n}{3}$ чисел. Вероятность попасть в B равна $\frac{n/3}{n} = \frac{1}{3}$.

Надежный способ.
Найдем статистики с номерами $\frac{n}{3}$ и $\frac{2n}{3}$, берем всё между ними.

6. **Приближение** MAX-SAT

Детерминированный способ.
Возьмём произвольные значения переменных, если выполнено меньше $\frac{m}{2}$ клозов, инвертируем все переменные. Итого: мы выполнили $\geqslant \frac{m}{2} \geqslant \frac{1}{2}\texttt{OPT}$ клозов.

Рандомизированный способ. Подставим все переменные случайно.
Если в клозе k литералов, он выполнился с вероятностью $Pr = (1-2^{-k})$ (из 2^k вариантов нам не подходит ровно 1). Заметим, что $k \geqslant 1 \Rightarrow Pr \geqslant \frac{1}{2}$. E[количества выполненных клозов] $= \sum((1 - 2^{-k_i})) \geqslant \frac{1}{2}m \geqslant \frac{1}{2}\texttt{OPT}$.

Но нам нужно не матожидание, а гарантия. Тут поможет предыдущая задача.
Максимизировать число выполненных клозов $\Leftrightarrow$ минимизировать число нарушенных.
Если отклонились от X менее, чем на $\frac{1}{2}$, то из-за целочисленности попали в X.
$\frac{1}{2}m + \frac{1}{2} = \frac{1}{2}m(1 + \frac{1}{m})$. Пользуемся предыдущей задачей для $\varepsilon = \frac{1}{m}$, нужно

$$\frac{1+\varepsilon}{\varepsilon} = \frac{1+1/m}{1/m} = m + 1$$

запусков.

7. **Random в задачах оптимизации**

Алгоритм: запустить много раз, выбрать наименьший ответ.

По неравенству Маркова на одном запуске ответ будет $> C(1+\varepsilon)\texttt{OPT}$ с вероятностью $< \frac{1}{1+\varepsilon} = 1 - \frac{\varepsilon}{1+\varepsilon}$ (это вероятность ошибки).

Обозначим $p = \frac{\varepsilon}{1+\varepsilon}$, после $\frac{1}{p} = \frac{1+\varepsilon}{\varepsilon}$ запусков Pr[ошибки] $= (1-p)^{\frac{1}{p}} \leqslant e^{-1}$.

8. **Первообразный корень**

Идея. Ткнём в случайное число из $[2, p-1]$ и проверим. Будем тыкать, пока не найдем.

Оценка. g – первообразный $\Rightarrow \forall k\colon (k, p-1) = 1$ g^k тоже первообразный.
Итого первообразных корней $\varphi(p-1) \Rightarrow$ вероятность попасть $\frac{\varphi(p-1)}{p-1} \geqslant \frac{1}{\log\log p}$.

Проверка числа g. Достаточно убедиться, что $x^{(p-1)/d} \neq 1$ для всех простых $d \mid (p-1)$.
Для этого один раз факторизуем $p-1$ и каждую проверку будем делать за $\mathcal{O}(\log^2 p)$. Факторизовать $p-1$ сейчас умеем за $\mathcal{O}(p^{1/4} \log p)$.

Итого. $\mathcal{O}(p^{1/4} \log p + k \log^2 p \log\log p)$ для вероятности ошибки e^{-k}.

9. (*) **Проверка ДЗ**

$n{=}2k$. Если мы проверяем первую половину дз, то в каждой из оставшихся задач $\frac{1}{3}$ ошибок $\Rightarrow \mathsf{E} = \frac{1}{3} \cdot k$, вероятность наличия ошибок $1 - \left(\frac{2}{3}\right)^k$.

Если мы проверяем случайную половину, ровно также. Непроверенные биты всё равно не зависят от проверенных.

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-COLORING $\to$ 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:

a) $\forall$ 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.
b) $\forall$ 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.

12. (*) DOMINATING-SET $\in$ NPc: VERTEX-COVER $\to$ 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.

13. (*) STEINER-TREE $\in$ NPc: SET-COVER $\to$ 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

8 обязательных задач, не забудьте глянуть вторую страницу дз!

1. **(2) Понижение вероятности**

   Алгоритм работает за $\mathcal{O}(n^3)$, вероятность **успеха** $1/n^3$ (ошибка односторонняя).
   За какое время можно добиться вероятности успеха $1 - 1/2^n$?

2. **(2) Randomized NP**

   Что будет, если в NP проверяющий подсказку алгоритм будет из ZPP, а не из P? Как полученный класс соотносится с уже известными?

   Подсказка: покажите, что этот класс содержит NP. *Задачу сдавайте до мягкого дедлайна так как, опыт прошлых поколений показывает вероятность ошибок в решении* $\approx 100\%$.

3. **(2) Сдать Полларда**

   В системе $C_1 = 100$ тестов. Поллард работает с вероятностью $\frac{1}{2}$.
   Сколько раз нужно запустить Полларда, чтобы с вероятностью $1 - C_2 = 1 - \frac{1}{10}$ получить `OK`?
   Больше баллов получат простые вычисления, которые можно сделать в уме для $\forall\ C_1, C_2$.

4. **(2)** MAX-CUT.

   Search-версия MAX-CUT: дан неорграф с весами w_e на рёбрах. Разрез (A, B) — это разбиение вершин V на две части: $V = A \sqcup B$ и $A, B \neq \varnothing$ (части не обязательно связные).
   Величина разреза $w(A, B) = \sum\limits_{\substack{a \in A, b \in B \\ (ab) \in E}} w_{ab}$. Найти разрез, для которого $w(A, B)$ максимально.

   Сформулируйте decision-версию задачи и решите search-версию через неё.
   Нужно найти величину разреза и сам разрез.

5. **(1)** Subgraph Isomorphism $\in$ NPc.

   Мы обсуждали, что задача Graph Isomorphism (GI) считается не NP-трудной.
   Subgraph Isomorphism: даны неорграфы G и H, определить, $\exists$ ли в G подграф $G' : G' \simeq H$.
   Напомним, что G' называется подграфом G, если он получен из G выкидыванием каких-то вершин и рёбер. Докажите, что Subgraph Isomorphism $\in$ NPc.

6. **(2)** EXACT-SET-COVER $\to$ SUBSET-SUM.

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

7. **(2)** SUBSET-SUM → PARTITION → JOB-SHEDULING

   PARTITION: разделить множество предметов на два, равных по суммарному весу.
   JOB-SHEDULING: даны n заказов, у каждого дано время выполнения t_i. Есть k одинаковых рабочих, распределить заказы по рабочим: время завершения всех работ → min.

8. **(2) Приближённый** MAX-3-SAT

   Придумайте приближенный ZPP алгоритм с константой лучше $\frac{3}{4}$ для MAX-3-SAT, про который дополнительно известно что в каждом клозе *ровно три различные переменные*.

3.2. Дополнительная часть

1. **(3)** MAX-2-SAT ∈ NPc
2. **(3+2) Полиномиальные нижние оценки!**
   a) **(3)** Покажите SETH ⇒ нельзя найти доминирующее множество размера k за $\mathcal{O}(n^{k-\varepsilon})$.
   b) **(2)** Покажите SETH ⇒ нельзя найти доминирующее множество размера k за $\mathcal{O}(n^{k-1-\varepsilon})$ в графах с маленькой средней степенью ($m = \mathcal{O}(n\text{polylog}(n))$).