Алгоритмы и модели вычислений. Контрольная 1. 27 марта 2023 ♣

Фамилия: Ковалёв *Группа:* Б05-105

Преподаватель, ведущий занятия:

Шестаков

- Заполните поля Фамилия, группа и имя преподавателя на обложке.
- Ответы без обоснования не оцениваются.
- При написании контрольной можно пользоваться любыми печатными/рукописными материалами.
- Нельзя пользоваться никакими электронными приспособлениями в том числе для просмотра конспектов или литературы.
- Нельзя пользоваться помощью других студентов, в том числе нельзя передавать печатные или рукописные материалы друг другу во время написания контрольной.
- Пользование электронными приспособлениями, помощью других студентов и списывание приравнивается к нечестному написанию контрольной и наказывается на усмотрение проводящих контрольную вплоть до обнуления работы.
- Всюду, если только явно не указано обратное, предполагается истинность стандартных гипотез $\mathcal{P} \neq \mathcal{NP}, \mathcal{NP} \neq \mathrm{co}\mathcal{NP}$ и тому подобных.
- Можно без доказательства пользоваться результатами, доказанными и разобранными на лекциях и семинарах.
- "Граф" без дополнительных указаний – это всегда неориентированный граф без петель и кратных рёбер, "цикл" – это всегда простой цикл, "путь" – простой путь.
- После написания контрольной вы должны сфотографировать работу на телефон, сформировать .pdf файл, отправить этот файл на почту acm-mipt@yandex.ru, в теме письма указать фамилию и группу. Затем сдать бумажную работу преподавателю.
- Разбалловка контрольной работы будет сообщена после проверки всех работ.

(1.) Пусть M – ДМТ, пусть $L \subseteq \{0,1\}^*$ – язык, причём выполняются два пункта.

a) Для всех слов w, таких что $|w| \geqslant 2023$,

- M принимает слово w, если $w \in L$,
- M отвергает слово w, если $w \notin L$,
- M работает на слове w не более $|w|^{38}$ тактов.

b) На всех словах w, таких что $|w| < 2023$, ДМТ M не останавливается (работает бесконечно долго).

Верно ли, что $L \in \mathcal{P}$?

(2.) Пусть язык L состоит из всех двоичных записей чисел, имеющих вид $p_1^{p_2}$, где p_1, p_2 – простые числа. Верно ли, что существует сводимость $L \leqslant_p \mathrm{SAT}$?

(3.) В этой задаче предполагается, что $\mathcal{P} \neq \mathcal{NP}$.
Пусть a) $A \in \mathcal{NPC}$, b) $B \in \mathcal{P}$, c) $B \subset A$.
Докажите, что $A \setminus B \in \mathcal{NPC}$.

(4.) Назовём булеву формулу φ **разнообразной**, если для неё существует два набора аргументов:

a) набор, на котором φ истинна,

b) набор, на котором φ ложна.

Язык SAT± определяется так: SAT± = $\{\varphi : \varphi$ является разнообразной$\}$

1. Докажите что SAT± $\in \mathcal{NP}$-hard.
2. Пусть существует ДМТ (оракул) M, которая за один такт может вычислять принадлежность φ языку SAT±. Как в таком случае за полиномиальное время найти набор, на котором φ истинна, и набор, на котором φ ложна, или корректно ответить, что требуемой пары наборов не существует?

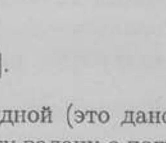
(5.) Тройкосочетанием в графе G называется набор вершинно непересекающихся треугольников в G. Тройкосочетание называется максимальным, если оно не является собственным подмножеством другого тройкосочетания в G. Размер тройкосочетания $|M|$ – это число треугольников в M.

1. Пусть M_1 и M_2 – два максимальных тройкосочетания в графе G. Докажите, что $|M_1| \leqslant 3|M_2|$.
2. Задача поиска наибольшего по размеру максимального тройкосочетания является $\mathcal{NP}$-трудной (это дано, не нужно это доказывать). Придумайте полиномиальный алгоритм, приближённо решающий эту задачу с погрешностью 3 (то есть, ваш алгоритм должен выдавать ответ равный не менее трети от оптимального ответа).

(6.) Пусть дан граф G, две его вершины назовём соседними, если они соединены ребром. Маршрут в графе G – это конечная последовательность попарно соседних вершин G (вершины и соответствующие рёбра могут появляться в маршруте не единожды). Замкнутый маршрут в G – маршрут, первая и последняя вершины которого совпадают (при этом эта вершина может присутствовать в маршруте более двух раз).
Назовём замкнутый маршрут в графе **почти гамильтоновым**, если он содержит каждую вершину графа ровно один раз, кроме одной вершины – её он может содержать более (но не менее) одного раза (совпадение первой и последней вершины в замкнутом маршруте не считается повторением).
Язык NHW состоит из описаний графов G таких, что в них существует почти гамильтонов замкнутый маршрут. Докажите $\mathcal{NP}$-трудность языка NHW.

(7.) Пусть для графа G величина $c(G)$ – это размер максимальной клики в G. Задача поиска Count-Max-Cliques состоит в следующем: дан граф G, найти **количество** клик размера $c(G)$.

Например, для графа $G_4 = (\{1,2,3,4\}, \{\{1,2\},\{2,3\},\{3,4\},\{4,1\}\})$, изображённого справа, величина $c(G_4)$ равна двум, а правильный ответ в задаче Count-Max-Cliques равен четырём: в G_4 есть четыре клики размера $c(G_4) = 2$, а именно $(\{1,2\},\{2,3\},\{3,4\},\{4,1\})$.



Пусть алгоритм A приближённо решает задачу Count-Max-Cliques в следующем смысле: если для графа G ответ на задачу Count-Max-Cliques равен x, то алгоритм A на графе G выдаёт число из отрезка $\left[\frac{x}{2}, 2x\right]$.
Докажите, что если $\mathcal{P} \neq \mathcal{NP}$, то A не может быть полиномиальным алгоритмом.

№ 1

построим новую машину M^*:
на входе w

1) если $|w| \geq 2023$, она просто запускает M и возвращает её ответ

2) ~~если $|w| < 2023$, то~~ в M^* записано про каждое слово длины < 2023, принадлежит ли оно языку. Это возможно, т.к. таких слов конечное число. Если $|w| < 2023$, то M^* возвращает заранее записанный в нее ответ

при этом M работает за полином от длины входа

$\Downarrow$

M^* тоже $\Rightarrow$ M^* - МТ, распознающая L за полином

$\Downarrow$

$L \in P$

ответ: да, верно.

№ 2

покажем, что язык $L \in NP$.

для него есть сертификат — $(p_1; p_2)$

длина сертификата $\log p_1 + \log p_2 = poly(|p_1| + |p_2|)$

посчитать $p_1^{p_2}$ можно за $O(\log p_2)$ умножений, каждое из которых требует $O(\log^2 p_1)$ операций

$\Downarrow$

$(p_1; p_2)$ действительно сертификат

т.е. $L \in NP \Rightarrow L \leq_p SAT$, т.к. $SAT \in$ NP-complete.

ч.т.д.

N 3

$A \in NP\text{-complete}$; $B \in P$; $B \subset A$

~~$\forall X \in NP$ $X \le_p A$ $\Rightarrow$ существует полиномиально вычислимая f_X, осуществляющая сводимость~~

~~пусть~~ $\tilde{f}_X(x) = \begin{cases} f_X(x), & x \notin B \\ y, & x \in B \end{cases}$, ~~где y – некоторый элемент из $\{A \setminus B\}$ (оно не пусто, т.к. $P \neq NP$)~~

~~$\tilde{f}_X$ полиномиально вычислима, т.к. проверка $x \in B$ полиномиальна, как и вычисление f_X~~

$\Downarrow$

~~$\tilde{f}_X$ сводит $X \le_p A \setminus B$~~

построим $A \le_p A \setminus B$; $f: A \to A \setminus B$

пусть $f(x) = \begin{cases} x, & x \notin B \\ y, & x \in B \end{cases}$, где y – некоторый элемент из $A \setminus B$ (оно не пусто, т.к. $P \neq NP$)

f определяет сводимость, т.к.

$x \notin A \Rightarrow x \notin B \Rightarrow f(x) = x \notin A \Rightarrow f(x) \notin A \setminus B$

$x \in A \Rightarrow \begin{cases} x \notin B \Rightarrow f(x) = x \in A \setminus B \\ x \in B \Rightarrow f(x) = y \in A \setminus B \end{cases}$

f полиномиально вычислима, т.к. проверка $x \in B$ полиномиальна, т.к. $B \in P$

$\Downarrow$

$\underset{\substack{\Uparrow \\ NP\text{-complete}}}{A} \le_p A \setminus B \Rightarrow A \setminus B \in NP\text{-complete}$ ч.т.д.

№ 4

1) $SAT \le_p SAT\pm$

пусть x не входит в φ

$\varphi \longmapsto \varphi \wedge x$

$\varphi \in SAT$ => существует выполняющий набор для φ =>

=> если к нему добавить $x = 0$, то $\varphi \wedge x = 0$

если к нему добавить $x = 1$, то $\varphi \wedge x = 1$ | => $\varphi \wedge x$ разнообразна, $\varphi \wedge x \in SAT\pm$

$\varphi \notin SAT$ => φ всегда 0 => $\varphi \wedge x$ всегда 0 =>

=> $\varphi \wedge x \notin SAT\pm$

$\Downarrow$

$SAT\pm \in NP\text{-hard}$

2) будем действовать так: (в формуле n переменных)

~~возвращаем~~

0) узнаем у оракула, разнообразна ли формула, если нет, то вернем нет и закончим: формула не в $SAT\pm$

1) $i = 0$

2) подставим вместо i-той переменной 1 (тождественно истинную формулу, например $x \vee \bar{x}$)

3) спросим у оракула, разнообразна ли новая формула:

3.1) если да, то $i += 1$ и вернемся к шагу 2

3.2) если нет, то подставим вместо переменных с $(i+1)$ до n что угодно и получим набор, на котором формула 0 или 1. Также подставим вместо i-той переменной 0, вместо остальных с $(i+1)$ по n что угодно и получим набор, на котором формула принимает оставшееся нужное нам значение. Вернем эти два набора и закончим.

Ясно, что алгоритм корректен и полиномиален, т.к. делает линейное число операций

N 5

Пусть M_1 и M_2 – max троекосочетания и $|M_1| > 3|M_2|$
тогда в M_1 больше треугольников, чем вершин в M_2
тогда хотя бы 1 треугольник из M_1, не содержит
ни одной вершины из M_2 $\Rightarrow$ в M_2 можно
добавить этот треугольник $\Rightarrow$ M_2 не max – противоречие

$\Downarrow$

$|M_1| \leq 3|M_2|$
ч.т.д.

2) всего в графе $C_n^3 = O(n^3)$ подграфов на 3 вершинах
будем брать произвольный, проверять, треугольник
ли он:

1) если да, то берем его в тройкосочетание M и
удаляем его из графа

2) если нет, просто переходим к следующему
делаем это, пока в графе есть треугольники
если в графе больше нет треугольников, то
полученное тройкосочетание M максимально по определению,
а значит по пункту 1 $|M| \leq 3|OPT|$, где OPT – троекосочетание max размера
$|OPT| \leq 3|M|$

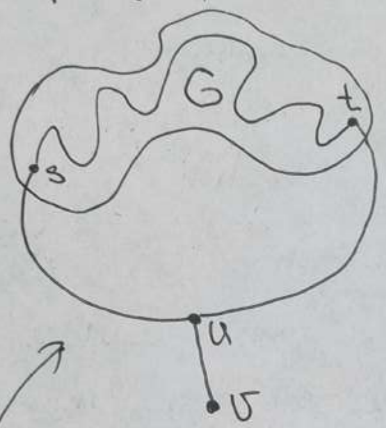
$\Downarrow$

$\frac{|M|}{|OPT|} \leq 3$; $\frac{|OPT|}{|M|} \leq 3$ $\Rightarrow$ алгоритм находит ответ
с погрешностью 3.

N6

s-t-HamPath $\leq_p$ NHW

$(G; s; t) \longmapsto G'$



в любом NHW в G' вершина u должна встречаться хотя бы дважды

пусть в G есть HamPath

$\Downarrow$

в G' есть NHW: $v-\underline{\underline{u}}-s-\ldots-t-$
$-\underline{\underline{u}}-v$,
в нем лишь u встречается дважды

~~пусть в G нет HamPath, а в G' есть NHW. В этом NHW вершина u встречается дважды => все остальные по 1 разу, но при этом нет пути из s в t, в котором все вершины встречаются по 1 разу => на любом гамильтоновом маршруте в G' хотя бы 2 вершины будут встречаться дважды => в G' нет NHW~~

пусть в G' есть NHW. В нем u встречается дважды => все остальные вершины по 1 разу => все вершины на пути s-t в G встречаются по 1 разу и идут друг за другом => в G есть HamPath.

$\Downarrow$

NHW $\in$ NP-hard.

№7 пусть алгоритм A полиномиален, $P \neq NP$

рассмотрим язык $L = \{(G; x)$: в G кол-во max клик лежит в $[\frac{x}{2}; 2x]\}$

тогда построим алгоритм, который за полином находит точное число max клик в G: $\hat{P}$

сначала поймем, в каком отрезке лежит число $(n = |V|)$ max клик: n раз узнаем, правда ли что в отрезке $[\frac{x}{2}; 2x]$ лежит число max клик: для этого возьмем $x = 2; 10; \dots$, т.е. имеющие вид $2 \cdot 4^a + 2$ по a от 0 до $n-1$.

в графе 2^n подграфов $\Rightarrow$ не более 2^n max клик $\Rightarrow$ $\Rightarrow$ мы точно узнаем, в каком отрезке лежит число клик. длина ~~этого~~ отрезка $\leq 2^n$.

Далее в нем опять с помощью алгоритма A узнаем, какое именно число max клик в G: для ~~этого~~ можно воспользоваться бинпоиском: пусть l - середина этого отрезка, тогда спросив у A ~~...~~ $(G; 2l)$ узнаем, в какой половине отрезка число max клик.

Итого сначала $n \cdot poly(n)$ операций, затем еще $\log 2^n \cdot poly(n) =$ $= n \cdot poly(n)$ операций, т.е. мы за полином определили, сколько точно в G max клик.

Но задача $\text{CountMaxClique} = \{(G; k)$: в G k max клик$\}$ $\hat{}$ NP-hard

если в графе есть хотя бы 2 клики max размера K_1 и K_2, то $\exists u \in K_1, u \notin K_2$, $\exists v \in K_2, v \notin K_1$ ...

$\Downarrow$

любая NP задача решается за полином $\Rightarrow$ $P = NP$ противоречие

$\Downarrow$

не существует такого полиномиального алгоритма

ч.т.д.