Программа курса

0) К началу курса необходимо иметь понятие о моделях вычислений в принципе. Необходимы знания про машины Тьюринга (ТФСиА), про алгоритмически разрешимые и неразрешимые задачи (ТФСиА), про детерминированность и недетерминированность (ТРЯП), про простые модели вычислений, которые не способны решать некоторые имеющие решения задачи (ТРЯП). Кроме того, необходимо владеть базовыми понятиями об алгоритмах вообще (ОА), об асимптотическом времени их работы (ОА), об алгоритмах обхода графов (ОА) и также иметь общее представление о модульной арифметике, теореме Эйлера и тому подобном (ОВАиТК).

1) Детерминированная и недетерминированная машина Тьюринга (ДМТ и НМТ). Понятие о вычислении на ДМТ и НМТ, понятие о распознавании языка на ДМТ и НМТ. Временная и пространственная сложность вычислений. Теорема об ускорении. Теоремы об иерархии. Универсальная МТ.

2) Классы DTIME, NTIME, их связь. Классы $\mathcal{P}$ и $\mathcal{NP}$, два определения класса $\mathcal{NP}$, их эквивалентность. $\mathcal{NP}$-трудные и $\mathcal{NP}$-полные языки, связанные с этим задачи. Предположительная трудность $\mathcal{NP}$-трудных задач.

3) Сводимость по Тьюрингу, по Куку и по Карпу. Использование сводимости по Карпу для доказательства $\mathcal{NP}$-полноты. Теорема Кука-Левина.

4) Со-классы, классы со$\mathcal{P}$, со$\mathcal{NP}$, их связь с $\mathcal{P}$ и $\mathcal{NP}$, примеры задач из класса со$\mathcal{NP}$. Полиномиальная иерархия и вычисления с оракулом. Классы $\mathcal{PH}$ и $\mathcal{EXP}$.

5) Классы DSPACE, NSPACE, их связь с временными классами. Класс $\mathcal{L}$, примеры задач из этого класса. Класс $\mathcal{PSPACE}$. Теорема Савича.

6) Класс $\mathcal{NL}$, логарифмическая сводимость. $\mathcal{NL}$-полнота. Теорема Иммермана-Селепченьи.

7) Неравномерная и схемная сложность. Класс сложности $\mathcal{P}$/poly. Распознавание языка семейством схем, класс SIZE.

8) Вероятностная машина Тьюринга, понятие о вероятностном вычислении. Классы $\mathcal{RP}$ и со$\mathcal{RP}$, их связь с $\mathcal{NP}$ и со$\mathcal{NP}$. Примеры языков из классов $\mathcal{RP}$ и со$\mathcal{RP}$.

9) Классы $\mathcal{BPP}$ и $\mathcal{PP}$, их связь и связь с невероятностными классами. Дерандомизация и независимость классов от точных значений числовых границ в классах.

10) Основания криптографии. Описание схемы RSA.

11) Алгоритм быстрого преобразования Фурье, его применение.

Литература

1. “Computational Complexity: A Modern Approach” by Sanjeev Arora and Boaz Barak
2. “Computational Complexity” by Christos Papadimitriou
3. “Computational Complexity: A Conceptual Perspective” by Oded Goldreich
4. “Complexity of Algorithms” by László Lovász
5. “Introduction to the Theory of Computation” by Michael Sipser
6. “The Nature of Computation” by Cristopher Moore and Stephan Mertens
7. “Mathematics and Computation” by Avi Wigderson

8. “Computers and Intractability: A Guide to the Theory of NP-Completeness” by Michael Garey and David Johnson

9. “Сложность вычислений” Мусатов Д.В., черновик

Задание

1.1. Дать максимально точные асимптотические оценки сверху и снизу для $T(n)$ (считать $T(n)$ константой для $n \leqslant 3$).

а) $T(n) = T(\frac{n}{2}) + T(\frac{n}{3}) + T(\frac{n}{6}) + cn$,

б) $T(n) = T(\frac{n}{2}) + T(\frac{n}{4}) + T(\frac{n}{6}) + cn$,

в) $T(n) = T(\sqrt{n}) + 1$.

1.2.

1) Покажите, что язык PAL = $\{w \mid w \in \{0,1\}^*, w$ – палиндром$\}$ распознается за $T(n) = O(n)$ на многоленточной МТ.

2) Докажите, что любая одноленточная МТ, распознающая PAL, работает за время $T(n) = \Omega(n^2)$

1.3. Пусть M работает за время $T(n)$ и существует такое n_0, что $T(n_0) < n_0 + 1$. Докажите, что:

1) M не читает символ номер $n_0 + 1$ ни у какого входа;

2) M работает за время $O(1)$;

3) M распознаёт регулярный язык.

1.4. Постройте одноленточную МТ, которая переводит вход 1^n в выход 1^{2n}, т.е. удваивает слово, заданное в унарном алфавите. Время работы $O(n \log n)$.

1.5. Предъявите какую-нибудь эффективную (вычислимую и при этом желательно быструю) процедуру получения строки над алфавитом $\{0,1\}$ по объекту и наоборот получения объекта по строке для следующим множеств объектов. Заметьте, что декодирование (получение объекта по строке) может быть сопряжено с технической трудностью: не все строки при кодировании становятся образом какого-то объекта – среди строк появляется “мусор”.

Объекты:

а) пары бинарных строк с разделителями;

б) натуральные числа;

в) булевы формулы;

г) рациональные числа (можно ли вещественные?);

д) матрицы;

е) графы;

ё) машины Тьюринга.

2.1. Доказать, что класс $\mathcal{P}$ замкнут относительно операций:

– объединения,

– пересечения,

– конкатенации,

– итерации (звёздочки Клини),

– дополнения,

– чётной итерации.

2.2. Доказать что языки принадлежат $\mathcal{P}$:

– язык всех несвязных графов без циклов (кодировка графа на ваше усмотрение),

– язык всех квадратных матриц на целых числах, в которых есть подматрица размера 2022×2022, заполненная нулями,

– язык всех графов, в которых существует эйлеров цикл и как минимум пять вершин имеют нечётную степень.

2.3.

1) доказать эквивалентность двух определений класса $\mathcal{NP}$,

2) доказать, что $\mathcal{P} \subset \mathcal{NP} \cap \mathrm{co}\mathcal{NP}$,

3) доказать, что $\mathcal{P} = \mathcal{NP}$ влечёт $\mathcal{NP} = \mathrm{co}\mathcal{NP}$,

4) доказать, что $\mathrm{co}\mathcal{NP} \neq \left(2^{\Sigma^*} \setminus \mathcal{NP}\right)$,

5) доказать, что $\mathcal{NP} \subset \bigcup_{k=0}^{\infty} \mathrm{DTIME}\left(2^{n^k}\right)$.

2.4. Пусть $L \in \mathrm{co}\mathcal{NP}$, тогда $\bar{L} \in \mathcal{NP}$ по определению. Тогда для $\bar{L}$ есть МТ M такая, что $M(x,u) = 1$ для $x \in \bar{L}$ и $M(x,u) = 0$ для $x \notin \bar{L}$. Но тогда можно определить новую МТ M', поменяв состояния M на противоположные. Тогда $M'(x,u) = 1$ для $x \notin \bar{L}$, т.е. для $x \in L$ и наоборот. Тогда $L \in \mathcal{NP}$ и $\mathcal{NP} = \mathrm{co}\mathcal{NP}$.

Где ошибка в рассуждениях?

2.5. Всюду под «можно ли» подразумевается следующее: останется ли при такой замене класс языков $\mathcal{NP}$ неизменным.

Рассмотрим следующее определение класса $\mathcal{NP}$: $L \in \mathcal{NP}$ означает существование полиномиальной по длине входа $|x| + |y|$ машины Тьюринга M, что верно

$$(x \in L) \Leftrightarrow \exists y \left(y \in \Sigma^{q(|x|)} \wedge M(x,y) = 1\right)$$

1) В определении стоит $y \in \Sigma^{q(|x|)}$. Можно ли смягчить требование, положив $y \in \Sigma^*, |y| \leqslant q(|x|)$?

2) В определении МТ останавливается за $p(|x| + |y|)$ шагов. Можно ли убрать из требования y, оставив $p(|x|)$ шагов?

3) Можно ли после предыдущих упрощений вообще убрать полиномиальное ограничение на y, получив такое определение

$L \in \mathcal{NP}$, если: существует ДМТ M, существует полином $p(n)$, такие, что

для любых $x \in \Sigma^*$, $y \in \Sigma^*$ машина M останавливается на входе (x,y) не более, чем за $p(|x|)$

шагов и

$$(x \in L) \Leftrightarrow \exists y\,(y \in \Sigma^* \wedge M(x,y) = 1)$$

2.6.

1) Привести пример языка L такого, что он не принадлежит $\mathcal{P}$.

2) Привести пример языка L такого, что он не принадлежит $\mathcal{P}$, а L^* принадлежит $\mathcal{P}$.

2.7. Доказать что языки принадлежат $\mathcal{NP}$.

1) Язык пар графов (G, H) таких, что графы G и H изоморфны.

2) Язык всех графов, содержащих эйлеров путь.

3) Язык всех графов, содержащих гамильтонов путь.

4) Язык всех взвешенных орграфов, в которых нет цикла отрицательной длины.

5) Язык всех составных чисел (записаны в десятичной системе счисления).

6) CNF–SAT

2.8. Для принадлежности классу $\mathcal{NP}$ не требуется никакого алгоритма получения сертификата, лишь его существование. Почему в таком случае каждый разрешимый язык L не принадлежит $\mathcal{NP}$, ведь для любого слова x можно предъявить очень простой сертификат: 1, если $x \in L$, или 0, если $x \notin L$?

2.9. Проблема останова МТ неразрешима. Однако, для слов, принадлежащих соответствующему языку, существует простейший сертификат – для пары (МТ, слово) сертификатом будет число N – число шагов до останова. Для любой пары (МТ, слово), такой, что МТ останавливается на этом слове, сертификат очевидно существует. Для любой пары, не принадлежащей языку, никакой сертификат не подойдёт. Следовательно, проблема останова принадлежит классу $\mathcal{NP}$, а значит разрешима на НМТ, т.е. разрешима. Где ошибка в рассуждениях?

2.10. Докажите, что классы $\mathcal{NP}$ и co$\mathcal{NP}$ замкнуты относительно операций:

а) объединения,

б) пересечения,

в) конкатенации,

г) итерации (звёздочки Клини).

2.11. Язык PRIMES — это язык простых чисел, двоичная запись натурального числа x принадлежит языку PRIMES тогда и только тогда, когда x простое число. Доказать, что

– PRIMES $\in$ co$\mathcal{NP}$,

– PRIMES $\in \mathcal{NP}$.

2.12. Докажите, что язык

FACTOR = $\{(n,k) \mid n$ содержит делитель, больший 1, но не превосходящий $k\}$

принадлежит классу $\mathcal{NP} \cap$ co$\mathcal{NP}$.

2.13. Докажите, что языки принадлежат co$\mathcal{NP}$

1) TAUT – язык, состоящий из описаний булевых тавтологий,

2) Язык, состоящий из пар (G, k) где G – описание графа, такого, что для любых k вершин найдется ребро, соединяющее хотя бы 2 из них.

3) Язык описаний графов, в которых есть клика на 2022 вершинах (клика – это подмножество вершин графа, таких, что каждая соединена ребром с каждой).

4) PLANARITY – язык описаний планарных графов.

3.1.

1) Верно ли, что если $L \in \mathcal{NPC}$ и $L \in \text{co}\mathcal{NP}$, то $\mathcal{NP} = \text{co}\mathcal{NP}$?

2) Верно ли, что если $L \in \mathcal{NP}$ и $L \in \text{co}\mathcal{NPC}$, то $\mathcal{NP} = \text{co}\mathcal{NP}$?

3) Верно ли, что язык $L \in \mathcal{NPC}$ тогда и только тогда, когда $\overline{L} \in \text{co}\mathcal{NPC}$?

4) Доказать, что если $\mathcal{P} = \mathcal{NP}$, то любой нетривиальный язык $\mathcal{NP}$-трудный. (Что насчёт тривиальных языков?)

5) Верно ли, что если $L_1 \leqslant_p L_2$, то $\overline{L}_1 \leqslant_p \overline{L}_2$?

3.2. По аналогии с обычной SAT определим задачу DNF–SAT: дана ДНФ, нужно проверить, выполнима ли она (т.е. дан язык всех выполнимых ДНФ, нужно построить распознаватель для него)

1) Придумать полиномиальный алгоритм решения DNF–SAT (это очень просто).

2) Построим сводимость CNF–SAT к DNF–SAT: в формуле КНФ раскрываем скобки в силу дистрибутивности операций конъюнкции и дизъюнкции. Доказать корректность сводимости.

3) Поскольку мы свели $\mathcal{NP}$–полную задачу к полиномиально разрешимой, то $\mathcal{P} = \mathcal{NP}$. Есть ли ошибка в рассуждениях?

3.3. Построить полиномиальную сводимость:

а) Hampath к Hamcycle

б) Hamcycle к Hampath

3.4. Мы рассматривали неориентированные графы на гамильтоновость. Рассмотрим теперь гамильтоновы ориентированные графы. Всюду приставка Dir означает ориентированность графа.

Построить полиномиальную сводимость:

а) Hampath к DirHampath

б) DirHampath к Hampath

Можете повторить то же самое для циклов.

3.5. Для каждого натурального k определим язык k–Clique всех графов, в которых есть клика размера k.

1) Язык 1–Clique полиномиален?

2) Язык 2–Clique полиномиален?

3) Язык 3–Clique полиномиален?

4?) Для каких k язык k–Clique $\mathcal{NP}$-полон?

5??) Известно, что Clique $\in \mathcal{NPC}$, но из входа (G, k) можно сперва выделить k, а затем запустить алгоритм решения соответствующего k–Clique. Как тогда может быть, что Clique $\in \mathcal{NPC}$?

6!) Доказать, что язык графов, имеющих клику ровно на половине вершин, $\mathcal{NP}$-полон.

3.6. Доказать, что следующие языки принадлежат $\mathcal{NPC}$.

1) Графы, имеющие вершинное покрытие ровно на половине вершин.

2) Пары граф G и число k такие, что существует простой путь в графе G длины как минимум

k.

3) Графы G, в которых есть простой путь, проходящий ровно по разу через все, кроме 2022 вершин.

4) Графы G, в которых есть минимальное остовное дерево со степенью всех вершин не более 42.

3.7. Описать полиномиальные алгоритмы для решения задач:

– 2–Partition

– 2–Color

– 2–SAT

3.8. Докажите, что язык Taut всех булевых тавтологий со$\mathcal{NP}$-полон. (Можно ограничиться формулами в ДНФ.)

3.9. Напомним, задача останова Halt неразрешима.

1) Верно ли, что Halt $\in \mathcal{NPC}$?

2) Верно ли, что Halt $\in \mathcal{NP}$-hard?

3.10. Замкнут ли $\mathcal{NPC}$ относительно:

а) объединения,

б) пересечения,

в) конкатенации,

г) итерации (звёздочки Клини)?

д) Если $L_1 \in \mathcal{NPC}$ и $L_2 \in \mathcal{NPC}$, то верно ли, что $L_1 \times L_2 \in \mathcal{NPC}$?

3.11. Большому количеству $\mathcal{NPC}$ задач и, соответственно, $\mathcal{NPC}$ языков соответствуют задачи поиска. Приведём пример:

Clique = $\{(G, k) \mid G$ содержит клику размера $k\}$ – задача распознавания и одновременно язык. Дана пара (G, k), выясним «да, принадлежит» или «нет, не принадлежит».

ExactClique = $\{(G, k) \mid G$ содержит клику размера k и не содержит клику размера $k + 1\}$ – тоже задача распознавания и одновременно язык. Дана пара (G, k), выясним «да, принадлежит» или «нет, не принадлежит».

MaxSizeOfClique$(G) = k$ – задача поиска, на вход подаётся граф G, на выходе нужно получить размер максимальной клики в графе G – число k.

MaxClique$(G) = G'$ – задача поиска, на вход подаётся граф G, на выходе нужно получить клику максимального размера, являющуюся подграфом G – назовём её G'.

Рассмотрим задачи Clique, ExactClique, MaxSizeOfClique, MaxClique. Докажите, что если для одной из них есть полиномиальный алгоритм, то он есть для каждой из этих задач. Не забывайте, что последние две не являются задачами распознавания – полиномиальность алгоритма для них определяется естественным способом.

3.12. Покажите, что если 3–Color $\in \mathcal{P}$, то за полиномиальное время можно не только определить, что граф допускает раскраску вершин в три цвета, но и найти какую-то 3-раскраску, если она существует.

4.1. Доказать, что языки лежат в $\mathcal{L}$

– ADD = $\{a\#b\#c \mid c = a + b\}$,

– MULT = $\{a\#b\#c \mid c = a \cdot b\}$,

– язык всех неориентированных графов, содержащих цикл,

– язык всех неориентированных графов, не содержащих цикл.

4.2. Доказать, что в $\mathcal{L}$ лежат

– язык правильных скобочных выражений,

– язык правильных скобочных выражений с двумя типами скобок,

– язык всех палиндромов.

4.3.

1) Доказать, что два определения класса $\mathcal{NL}$ эквивалентны.

2) Какой класс получится, если заменить полиномиальную длину сертификата на логарифмическую и убрать ограничение на чтение сертификата единожды?

3) Какой класс получится, если заменить полиномиальную длину сертификата на логарифмическую, но оставить ограничение на чтение сертификата единожды?

4) Какой класс получится, если оставить полиномиальную длину сертификата и убрать ограничение на чтение сертификата единожды?

4.4. Пусть $L_1 \leqslant_{\log} L_2$, $L_2 \leqslant_{\log} L_3$

1) доказать, что если $L_2 \in \mathcal{L}$, то $L_1 \in \mathcal{L}$,

2) доказать, что если $L_2 \in \mathcal{NL}$, то $L_1 \in \mathcal{NL}$,

3) доказать, что $L_1 \leqslant_{\log} L_3$.

4.5. Доказать, что stCON лежит в $\mathcal{NL}$

4.6. Доказать, что для любой "хорошей" f выполняется

$$\mathrm{DTIME}(f) \subset \mathrm{NTIME}(f) \subset \mathrm{DSPACE}(f) \subset \mathrm{NSPACE}(f) \subset \mathrm{DTIME}\left(2^{O(f)}\right)$$

4.7. Известно, что USTCON $\in \mathcal{L}$. Для каждого из языков доказать, что он лежит в $\mathcal{L}$ и что он эквивалентен по сложности USTCON (язык сводится к USTCON и обратно USTCON сводится к языку)

– язык всех не двудольных графов,

– язык всех двудольных графов,

– язык пар граф G и ребро графа e, таких, что в G есть цикл, проходящий через e

4.8. Доказать принадлежность $\mathcal{NL}$

– языка двудольных графов,

– языка 2–Color.

4.9. Доказать $\mathcal{NL}$-полноту языков

– ShortestPath = $\{(G, k, s, t) |$ G – орграф, кратчайший путь из s в t в графе G имеет длину ровно $k\}$

– всех сильносвязных орграфов,

– всех орграфов, содержащих цикл,

– всех орграфов, не содержащих цикла (DAG),

– 2-SAT,

– всех пар (A, ω), где НКА A принимает слово ω.

5.1. Доказать, что

1) $\mathcal{P} \subset \mathcal{P}/\text{poly}$,

2) $\mathcal{P} \neq \mathcal{P}/\text{poly}$ и даже $\mathcal{P} \neq \mathcal{P}/1$,

3) $\mathcal{P}/\text{poly}$ содержит разрешимый язык, не принадлежащий $\mathcal{P}$.

5.2. Доказать, что если в определении $\mathcal{P}/\text{poly}$ ограничиться полиномиально вычислимыми α (т.е. такими, что существует полиномиальная ДМТ на входе n выдающая α_n), такое ограничение даст класс $\mathcal{P}$.

5.3. 1) Язык называется унарным, если все его слова имеют вид 1^k. Доказать, что любой унарный язык принадлежит $\mathcal{P}/\text{poly}$

2) Пусть язык L обладает следующим свойством: любые два слова x и y одинаковой длины либо оба принадлежат L, либо оба не принадлежат L. Доказать, что $L \in \mathcal{P}/\text{poly}$.

5.4. Доказать, что $\mathcal{P}/O(2^n) = \text{ALL}$ – множество всех языков.

6.1. Вы бросаете правильную монету (вероятности орла и решки одинаковы) до тех пор, пока не выполнится определённое условие. Для каждого условия ниже найдите математическое ожидание числа бросков монеты.

1) Пока не выпадет решка.

2) Пока не выпадет две решки.

3) Пока не выпадет и орёл, и решка.

4) Пока в двух последовательных бросках не выпадут две решки подряд.

5) Пока в двух последовательных бросках не выпадут орёл, затем решка.

6) Пока в четырёх последовательных бросках не выпадут последовательно решка-орёл-решка-орёл.

7) Пока в четырёх последовательных бросках не выпадут последовательно решка-решка-орёл-орёл.

8) Пока суммарно число выпавших орлов не превысит число выпавших решек.

9) Пока суммарно число выпавших орлов не будет равно числу выпавших решек.

10) Пока в n последовательных бросках не выпадут n решек подряд.

6.2. Рассмотрим следующий алгоритм равномерного перемешивания колоды из n карт (это значит, что вероятность получить любое конкретное перемешивание одинакова):

1) возьмём колоду и пронумеруем карты сверху вниз от 1 до n;

2) далее работаем по шагам, на каждом шаге:

2.1) берём карту с вершины стопки,

2.2) в оставшейся колоде из $n-1$ карты равновероятно выбираем случайное место для карты (n способов – на верх колоды или под какую-то из карт колоды),

2.3) вставляем взятую карту в выбранное случайное место (карт в колоде снова n);

3) алгоритм заканчивает свою работу после того, как цикл сработал на карте номер $n-1$, т.е. карта номер $n-1$ была взята сверху и вставлена в случайное место.

6.3. Вам дана нечестная монета – вы не знаете, насколько она нечестна, т.е. вы не знаете вероятности выпадения орла и решки при броске, вы знаете лишь, что обе эти вероятности отличны от нуля.

1) Придумать алгоритм, симулирующий бросок честной монеты – равновероятное получение единицы или нуля. В качестве единственного источника случайных значений вы можете использовать нечестную монету.

2) Вычислить математическое ожидание числа «бросков» нечестной монеты для вашего алгоритма.

Формально, вы моделируете бернуллиевскую случайную величину с параметром $1/2$ при помощи бернуллиевской случайной величины с неизвестным параметром $p \in (0,1)$.

6.4. Вам дана честная монета – вероятности выпадения орла и решки при броске одинаковы.

1) Придумать алгоритм, симулирующий бросок нечестной монеты с заданной вероятностью выпадения орла и решки. В качестве единственного источника случайных значений вы можете ис-

пользовать честную монету.

2) Вычислить математическое ожидание числа «бросков» честной монеты для вашего алгоритма.

Формально, вы моделируете произвольную бернуллиевскую случайную величину с параметром $p \in (0,1)$ при помощи бернуллиевской случайной величины с параметром $1/2$.

6.5. Вы с другом договорились о числе $p \in [0,1]$. После этого ваш друг выбирает случайное число – с вероятностью p он выбирает 1, с вероятностью $1-p$ он выбирает 0.

Вы хотите угадать это число и для этого вы выбираете число q и пишете программу, которая

1) с вероятностью q выбирает 1, с вероятностью $1-q$ выбирает 0,

2) проверяет, равно ли выбранное число загаданному.

Если равно, число найдено, если не равно, алгоритм повторяет шаги 1 и 2.

Рассмотрим две постановки задачи

а) после неудачной попытки угадать ваш друг снова выбирает случайное число с вероятностью p;

б) после неудачной попытки число, загаданное другом, не изменяется.

Какое q вам нужно выбрать, чтоб минимизировать число попыток вашего алгоритма в среднем?

6.6. Лемма Шварца-Зиппеля. Пусть $p \in F[x_1, x_2, \ldots, x_n]$ – ненулевой полином от n переменных степени $d \geqslant 0$ над полем F. Пусть S конечное подмножество F и пусть элементы $r_1, r_2, \ldots, r_n$ были выбраны из S равномерно и независимо друг от друга.

Тогда

$$\mathbb{P}[p(r_1, r_2, \ldots, r_n) = 0] \leqslant \frac{d}{|S|}$$

Доказать лемму. Решить с помощью леммы задачу PIT (polynomial identity testing). Дан полином p, верно ли, что после раскрытия всех скобок и приведения его к сумме мономов все коэффициенты перед мономами будут равны нулю? В данном случае “решить” – это “предъявить эффективный вероятностный алгоритм”.

7.1. Назовём классом $\mathcal{BPP}_{(\varepsilon_1,\varepsilon_2)}$ класс языков, распознаваемых ПВМТ M, причём

$x \in L \Rightarrow \mathbb{P}(M(x) = 1) \geqslant 1 - \varepsilon_1$,

$x \notin L \Rightarrow \mathbb{P}(M(x) = 0) \geqslant 1 - \varepsilon_2$.

Напомним, что $\mathcal{BPP} = \mathcal{BPP}_{(1/3,1/3)}$ по определению.

Доказать, что для любых $\varepsilon_1, \varepsilon_2 < 1/2$ выполняется $\mathcal{BPP} = \mathcal{BPP}_{(\varepsilon_1,\varepsilon_2)}$

7.2. Назовём классом $\mathcal{BPP}_p$ класс языков, распознаваемых ПВМТ M, причём существует положительный полином $p(n)$ и

$x \in L \Rightarrow \mathbb{P}(M(x) = 1) \geqslant \frac{1}{2} + \frac{1}{p(|x|)}$,

$x \notin L \Rightarrow \mathbb{P}(M(x) = 0) \geqslant \frac{1}{2} + \frac{1}{p(|x|)}$.

Доказать, что $\mathcal{BPP} = \mathcal{BPP}_p$

7.3. Доказать, что

a) $\mathcal{RP} \subset \mathcal{NP}$, $\mathrm{co}\mathcal{RP} \subset \mathrm{co}\mathcal{NP}$

b) $\mathcal{RP} \subset \mathcal{BPP}$, $\mathrm{co}\mathcal{RP} \subset \mathcal{BPP}$

c) $\mathcal{BPP} \subset \mathcal{PP}$, $\mathcal{NP} \subset \mathcal{PP}$

7.4. MAXCUT – это задача следующего вида: дан граф G, нужно найти разбиение вершин графа на два множества так, чтобы число рёбер, у которых концы лежат в разнах множествах, максимально.

1) Придумать простой вероятностный алгоритм решения MAXCUT.

2) Дерандомизировать алгоритм, используя метод условных матожиданий.

7.5. MAXSAT – это задача следующего вида: дана булева формула в виде КНФ, нужно найти набор переменных, на котором выполняется наибольшее возможное число дизъюнктов.

1) Придумать простой вероятностный алгоритм решения MAXSAT.

2) Дерандомизировать алгоритм, используя метод условных матожиданий.

7.6. Пусть $\mathcal{ZPP} = \mathcal{RP} \cap \mathrm{co}\mathcal{RP}$. Доказать, что следующие утверждения эквивалентны:

1) $L \in \mathcal{ZPP}$;

2) существует вероятностная машина Тьюринга, выдающая на слове x правильный ответ с вероятностью единица, при этом не *худшее* время работы у неё полиномиально, а **ожидаемое** время работы полиномиально;

3) существует ПВМТ, которая на слове x с вероятностью $1/2$ отвечает верно (правильно определяется принадлежность или непринадлежность слова языку), а с вероятностью $1/2$ отвечает "ответ неясен".

7.7. Пусть мы запускаем $\mathcal{PP}$ или $\mathcal{BPP}$ алгоритм t раз. $X_i = 1$, если ответ алгоритма на запуске i правильный и 0 в противном случае.

Предположим, после t запусков мы принимаем решение о принадлежности слова языку на основании "мнения большинства". На основе оценки Чернова докажите, что класс $\mathcal{BPP}$ "эффективен", а класс $\mathcal{PP}$ "не эффективен", т.е. что для получения малой вероятности ошибки (пусть полиномиально малой или экспоненциально малой) в первом случае требуется полиномиальное число запусков

алгоритма, а во втором полиномиального числа запусков может не хватить. (Полиномиальное число – как и время работы алгоритма – считается от длины входа).

7.8. Определим задачу EZE (evaluate to zero everywhere). Дан полином p от n переменных степени $d \geqslant 0$ над полем F, верно ли, что при любом выборе элементов $r_1, r_2, \ldots, r_n$ из поля F, значение $p(r_1, r_2, \ldots, r_n) = 0$?

Доказать, что EZE $\in \mathrm{co}\mathcal{NP}$-hard.

Ранее было доказано, что PIT эффективно разрешим. Следует ли из вышесказанного, что $\mathrm{co}\mathcal{RP} = \mathrm{co}\mathcal{NP}$?

7.9. Назовём классом $\mathcal{PP}_{\text{more equal}}$ класс языков, распознаваемых ПВМТ M, причём

$x \in L \Rightarrow \mathbb{P}(M(x) = 1) > 1/2$,

$x \notin L \Rightarrow \mathbb{P}(M(x) = 0) \geqslant 1/2$.

Докажите, что $\mathcal{PP}_{\text{more equal}} = \mathcal{PP}$.

7.10. Назовём классом $\mathcal{PP}_{\text{even more equal}}$ класс языков, распознаваемых ПВМТ M, причём

$x \in L \Rightarrow \mathbb{P}(M(x) = 1) \geqslant 1/2$,

$x \notin L \Rightarrow \mathbb{P}(M(x) = 0) \geqslant 1/2$.

Докажите, что $\mathcal{PP}_{\text{even more equal}} = 2^{\Sigma^*}$.

8.1. Пусть нужно передать сообщение x по незащищённому каналу. Предложите схему передачи сообщения, одновременно включающую в себя цифровую подпись, т.е. подтверждающую личность отправителя.

Проведите явно вычисления, зашифровывающие сообщение и получающие подпись и расшифровывающие сообщение с подтверждением подписи.

Сообщение $x = 67$ передаётся от агента A с набором ключей $(e_A, d_A, n_A) = (11, 107, 1247)$ агенту B с набором ключей $(e_B, d_B, n_B) = (7, 547, 1357)$.

8.2. Ослепление

Вы хотите, чтоб некто M подписал своей электронной подписью сообщение x. Однако, очевидно, вы не добьётесь результата, послав M сообщение x, поскольку оно выглядит подозрительно. Однако, пусть (e, n) открытый ключ M, а d – его секретный ключ (вам неизвестный).

1) Возьмём случайное число r по модулю n и составим сообщение $y = r^e x \pmod n$.

2) Предположим, что y выглядит достаточно невинно, для того, чтобы M согласился подписать y своей электронной подписью и переслать вам подписанную версию: s_y.

3) Если M подпишет сообщение, то как по подписанному сообщению s_y и известным вам данным получить правильную подпись на сообщение x?

8.3. Внутренняя атака на общий модуль

Пусть Билли, Вилли и Дилли используют один и тот же модуль в своих ключах RSA. Пусть Билли переслал Вилли сообщение x, после шифрования получился шифр z. Покажите, что система не является криптоустойчивой, а именно, что Дилли может, зная собственные ключи, все открытые ключи и шифр z, эффективно вычислить x.

8.4. Внешняя атака на общий модуль

Вы обнаружили, что агенты J и K используют один и тот же модуль в своих ключах RSA. Открытый ключ агента J есть (e_J, n), открытый ключ агента K есть (e_K, n). Агент Z пересылает агентам J и K одно и то же зашифрованное сообщение x без использования цифровой подписи. Вы перехватываете сообщения и обнаруживаете, что он послал c_J агенту J и c_K агенту K. Расшифруйте x.

1) Пусть $n = 407$, $e_J = 7$, $e_K = 11$, $c_J = 368$, $c_K = 389$ (n нельзя разложить на множители).

2) Что, если бы модули были равны $e_J = 21$, $e_K = 35$?

8.5. Малый открытый ключ

На практике процедура кодирования иногда должна выполняться устройствами с малой вычислительной мощностью – смарт-картами. В таком случае целесообразно использовать малый по величине открытый ключ. Зафиксируем $e = 3$ для всех вычислений – кодирование тогда делается быстро: всего два возведения в степень. При этом, казалось бы, устойчивость алгоритма ко взлому не страдает, ведь вычислить x, зная $x^3 \pmod N$, но не зная разложение N на простые множители, – весьма неэлементарная операция.

Однако, пусть Алиса пересылает Борису, Виктору и Дмитрию одно и то же сообщение x. Открытые ключи адресатов $(3, N_1)$, $(3, N_2)$ и $(3, N_3)$, пересылаемые сообщения тогда $c_i = x^3 \pmod{N_i}$.

0) Почему нельзя было выбрать $e = 2$?

1) Пусть модули N_i не попарно взаимнопросты. Покажите как злоумышленник Ева, получившая доступ к c_1, c_2, c_3, может восстановить x.

2) Пусть модули попарно взаимнопросты. Как, зная c_1, c_2, c_3, легко (то есть полиномиально) получить величину $x^3 \pmod{N_1 N_2 N_3}$?

3) Как по полученному числу из предыдущего пункта легко вычислить сообщение x?

9.1. Один раз сделать БПФ руками – для двух кубических полиномов. То же в поле остатков. Полиномы выберите сами – только не слишком простые.

9.2. Пусть дано два массива чисел длины n: $x = (x_0, x_1, \ldots, x_{n-1})$ и $y = (y_0, y_1, \ldots, y_{n-1})$.

Поэлементным произведением двух массивов называют массив $z = (z_0, z_1, \ldots, z_{n-1})$ длины n, такой что $z_i = x_i y_i$. Мы будем писать $z = x \cdot y$

Конволюцией или свёрткой этих массивов называют массив $z = (z_0, z_1, \ldots, z_{2n-1})$ длины $2n$, такой что

$$z_k = \sum_{i+j=k} x_i y_j = \sum_i x_i y_{k-i} \quad k \in \{0, \ldots 2n-1\} \quad \max\{0, k-n\} \leqslant i \leqslant \min\{k, n\}$$

Мы будем писать $z = x * y$

Циклической свёрткой этих массивов называют массив $z = (z_0, z_1, \ldots, z_{n-1})$ длины n, такой что

$$z_k = \sum_{i+j=k \pmod n} x_i y_j = \sum_{i+j=k} x_i y_j + \sum_{i+j=k+n} x_i y_j$$

Мы будем писать $z = x \circledast y$.

1) Доказать, что если массив $A = (a_0, a_1, \ldots, a_n)$ – коэффициенты полинома $a(x)$, массив $B = (b_0, b_1, \ldots, b_n)$ – коэффициенты полинома $b(x)$, то коэффициенты полинома $a(x)b(x)$ есть массив $A * B$.

2) Пусть $\mathbb{DFT}$ – это прямое, а $\mathbb{IDFT}$ – обратное дискретное преобразование Фурье. Докажите теорему о свёртке:

$$x \circledast y = \mathbb{IDFT}\left(\mathbb{DFT}(x) \cdot \mathbb{DFT}(y)\right)$$

3) Сведите задачу о нахождении свёртки двух массивов к задаче нахождения циклической свёртки массивов.

9.3. Даны два конечных множества $A \subseteq \mathbb{N}$ и $B \subseteq \mathbb{N}$. Опишите алгоритм для вычисления суммы Минковского $A + B = \{a + b \mid a \in A, b \in B\}$, за $O(M \log M)$, где M – максимум из чисел множества $A \cup B$.

9.4. Циркулянтной матрицей порядка n или просто циркулянтом называется квадратная матрица C, вида

$$C = \operatorname{circ}(c_0, \ldots c_{n-1}) = \begin{Vmatrix} c_0 & c_1 & c_2 & \ldots & c_{n-1} \\ c_{n-1} & c_0 & c_1 & \ldots & c_{n-2} \\ c_{n-2} & c_{n-1} & c_0 & \ldots & c_{n-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_1 & c_2 & c_3 & \ldots & c_0 \end{Vmatrix}$$

Система линейных уравнений $Cx = b$ для циркулянтов решается быстро с помощью преобразования Фурье.

1) Доказать, что собственный вектор матрицы C номер j равен $(1, \omega_j, \omega_j^2, \ldots, \omega_j^{n-1})$, где $j = \overline{0, \ldots, n-1}$, а $\omega_j = \exp\left(i\frac{2\pi j}{n}\right)$.

2) Доказать, что собственные значения циркулянта получаются с помощью преобразования Фурье первой строки матрицы.

3) Привести алгоритм вычисления ранга циркулянта.

4) Пусть n есть степень двойки, получить алгоритм решения $Cx = b$ за $O(n \log n)$.

9.5. Пусть задан текст T длиной n и шаблон P длиной m над алфавитом Σ.

1) С помощью быстрого преобразования Фурье получить алгоритм поиска шаблоне в тексте – т.е. всех номеров позиций в тексте, начиная с которых текст совпадает с шаблоном. Время работы $O((n+m)\log(n+m))$.

2) Поскольку $m \leqslant n$, предыдущая оценка есть $O(n \log n)$. Усовершенствовать алгоритм и улучшить оценку до $O(n \log m)$.

3) Усовершенствовать алгоритм для поиска по шаблону со специальным символом $*$, который может быть равен любому символу из алфавита Σ. Время работы $O(n \log m)$.

9.6. Пусть размер вектора не является степенью двойки, можно, конечно, "добить" исходный вектор нулями, но иногда это нежелательно, к примеру, нам нужно преобразование Фурье само по себе. Пусть дан вектор p длины n.

По определению

$$p_j^* = \sum_{k=0}^{n-1} p_k \cdot \omega_n^{kj},$$

главный трюк здесь – замена

$$kj = \frac{k^2}{2} + \frac{j^2}{2} + \frac{-(j-k)^2}{2}.$$

Получаем

$$p_j^* = \sum_{k=0}^{n-1} p_k \cdot \omega_n^{k^2/2} \cdot \omega_n^{j^2/2} \cdot \omega_n^{-(j-k)^2/2}$$

Сомножитель $\omega_n^{j^2/2}$ от k не зависит, оставшиеся сомножители переобозначим:

$$a_k = p_k \cdot \omega_n^{k^2/2}, \quad b_k = \omega_n^{-k^2/2}$$

В результате мы получаем выражение

$$p_j^* = \omega_n^{j^2/2} \sum_{k=0}^{n-1} a_k b_{j-k}$$

Покажите, как вычислить последнее выражение с помощью стандартного быстрого преобразования Фурье. Заметьте, что свёртка здесь ведётся в том числе по отрицательным индексам b_i, которые должны быть корректно определены.