КАЗАНСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ
ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И
ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ

М.Ф. НАСРУТДИНОВ

СБОРНИК ЗАДАЧ И УПРАЖНЕНИЙ ПО ДИСЦИПЛИНЕ "КОМПЬЮТЕРНАЯ МАТЕМАТИКА". ЧАСТЬ 2

УЧЕБНО-МЕТОДИЧЕСКОЕ ПОСОБИЕ

Казань — 2024

УДК 514

Печатается по решению Учебно-методической комиссии
Института информационных технологий и интеллектуальных систем
Казанского федерального университета
Протокол 5 от 24 июня 2024 г.

Научный редактор:
доктор физико-математических наук, профессор
кафедры цифровой аналитики и технологий искусственного интеллекта КФУ
Елизаров А.М.

Насрутдинов М.Ф.
Сборник задач и упражнений по дисциплине "Компьютерная математика". Часть 2. Учебно-методическое пособие / М.Ф. Насрутдинов — Казань: Казанский федеральный университет, 2024. — 37 с.

Учебно-методическое пособие предназначено для студентов первого курса Института информационных технологий и интеллектуальных систем Казанского федерального университета для проведения практических занятий по курсу "Компьютерная математика" во втором семестре.

Оглавление

Пособие представляет собой сборник задач для проведения семинарских занятий во втором семестре по дисциплине "Компьютерная математика".

В начале каждого параграфа приведены формулы, определения и другие краткие пояснения теории, необходимые для решения последующих задач.

В конце каждого параграфа (после черты) приведены задачи для домашних заданий или повторений перед контрольными работами.

1 Конечные детерминированные автоматы

Конечный автомат (finite-state machine = FSM) – это математическая абстракция, модель, которая в каждый конкретный момент времени может находиться только в одном из конечного числа состояний. Автомат умеет переходить из одного состояния в другое в ответ на данные, которые подаются на вход; изменение состояния называется переходом. Заданы одно начальное и множество финальных (конечных, допустимых) состояний.

Определение 1.1. Детерминированным конечным автоматом (ДКА) называется пятерка объектов

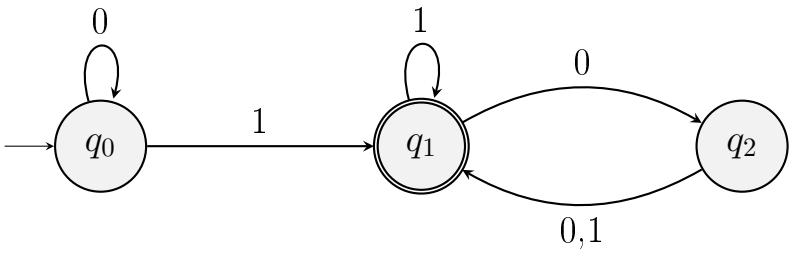
$$M = (Q, \Sigma, \delta, q_0, F),$$

где

1. Q – конечное множество состояний;
2. Σ – конечное множество символов (алфавит);
3. $\delta : Q \times \Sigma \to Q$ – функция перехода (обычно она записывается в виде таблицы);
4. $q_0 \in Q$ – начальное состояние;
5. $F \subset Q$ – множество финальных (допустимых) состояний.

Конечный автомат изображается графом (диаграммой состояний). Вершины обозначают состояния автомата, ребра – переходы из одного состояния в другое. Ребра помечены символами конечного алфавита.

Пример 1.1. Пусть задан конечный алфавит $\Sigma = \{0, 1\}$. Рассмотрим автомат M_1.



В этом автомате начальное состояние – q_0. Финальное состояние – q_1. Подадим на вход автомату слово ω над алфавитом Σ. Пусть $\omega = 1101$.

$$q_0 \xrightarrow{1} q_1 \xrightarrow{1} q_1 \xrightarrow{0} q_2 \xrightarrow{1} q_1.$$

Говорят, что автомат принимает слово, если после окончания работы он находится в одном из финальных состояний.

Можно убедиться, что автомат M_1 принимает слова (еще говорят – распознает) $1, 01, 11, 010101$ и $100, 0100$. В то же время автомат не принимает слова $0, 010$ или 101000.

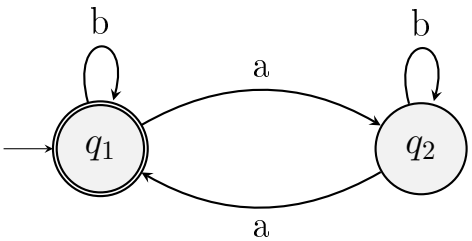
Формальное описание автомата $M_1 = (Q, \Sigma, \delta, q_0, F)$:

1. $Q = \{q_0, q_1, q_2\}$;
2. $\Sigma = \{0, 1\}$ – конечное множество символов (алфавит);
3. $\delta : Q \times \Sigma \to Q$ – функция перехода, заданная таблицей

δ	0	1
q_0	q_0	q_1
q_1	q_2	q_1
q_2	q_1	q_1

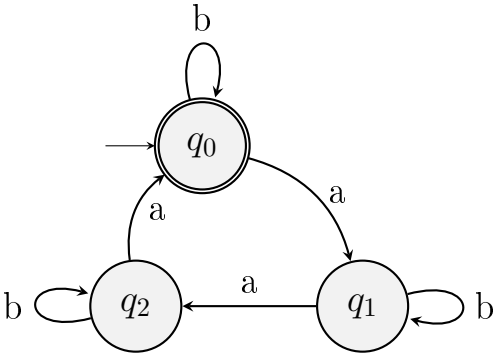
4. q_0 – начальное состояние;
5. $F = \{q_1\}$ – множество финальных (допустимых) состояний.

Пример 1.2. Пусть задан конечный алфавит $\Sigma = \{a, b\}$. Необходимо построить автомат M_2, распознающий четное число букв a (нуль тоже входит в этот набор).



Стратегия построения. Заведем два состояния: q_1 – четное количество, q_2 – нечетное количество, далее запишем переходы.

Пример 1.3. Пусть задан конечный алфавит $\Sigma = \{a, b\}$. Необходимо построить автомат M_3, распознающий слова, в которых число букв a делится на 3.



Пусть Σ – конечное множество символов (алфавит). Множество всех слов над алфавитом Σ обозначим Σ^*. **Языком** над алфавитом Σ называется произвольное множество слов, записанных буквами из Σ.

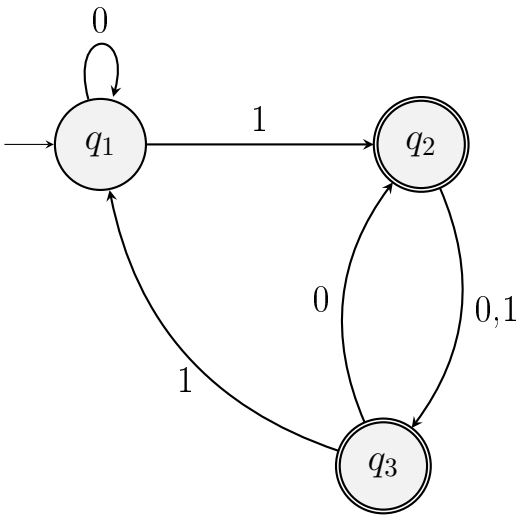
Определение 1.2. Пусть $M = (Q, \Sigma, \delta, q_0, F)$ – ДКА и $w = w_1 w_2 \ldots, w_n$ – строка длины n над алфавитом Σ, то есть $w_i \in \Sigma$. Говорят, что автомат M принимает слово w, если существует последовательность состояний $r_0, r_1, r_2, \ldots, r_n \in Q$, для которых выполнено

1. $r_0 = q_0$;
2. $\delta(r_i, w_{i+1}) = r_{i+1}, \qquad i = 0, 1, \ldots, n-1$;
3. $r_n \in F$.

Будем обозначать $L(M)$ язык, распознаваемый (принимаемый) автоматом M.

Задачи

1.1. Задан конечный автомат. Дайте формальное описание автомата и ответьте на следующие вопросы:



a) Какое состояние начальное? b) Какие состояния финальные? c) Принимает ли автомат строку 001001?

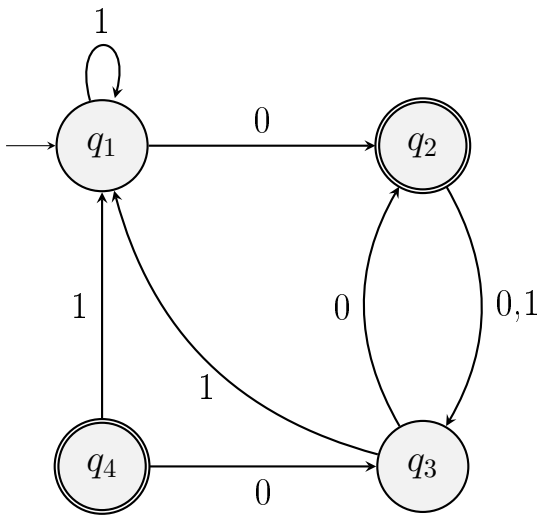
1.2. Постройте ДКА для языков над алфавитом $\Sigma = \{0, 1\}$. Здесь $\#w$ означает длину слова w, ε – пустое слово.

1. $L = \{w \mid w = 0w'1\}$;
2. $L = \{w \mid w$ содержит не менее трех $1\}$;
3. $L = \{w \mid w = w'0101w''\}$;
4. $L = \{w \mid w = xy0w''\},\ x, y \in \Sigma$.

1.3. Каждый из языков упражнения является пересечением более простых языков. Постройте ДКА для каждой простой части, а затем – автомат для исходного языка. Алфавит $\Sigma = \{0, 1\}$.

1. $L = \{w \mid w$ содержит не менее трех $1,$ и содержит не менее двух $0\}$;
2. $L = \{w | w$ содержит ровно две 1 и не менее двух $0\}$;
3. $L = \{w \mid w = 0w'$ и $\#w = 2k$ или $w = 1w'$ и $\#w = 2k + 1\}$.

1.4. Задан конечный автомат. Дайте формальное описание автомата и ответьте на следующие вопросы:



a) Какое состояние начальное? b) Какие состояния финальные? c) Принимает ли автомат строку 001001?

1.5. Постройте ДКА для языков над алфавитом $\Sigma = \{0, 1\}$. Здесь $\#w$ означает длину слова w, ε – пустое слово.

1. $L = \{w \mid w$ не содержит подслов $110\}$;
2. $L = \{w \mid$ длина $w \leq 5\}$;
3. $L = \{w \mid$ все слова кроме $11, 111\}$;
4. $L = \{w \mid w$ не содержит строки $01\}$;
5. $L = \{\varepsilon\}$.

1.6. Постройте ДКА для языков над алфавитом $\Sigma = \{0, 1\}$. Здесь $\#w$ означает длину слова w, ε – пустое слово.

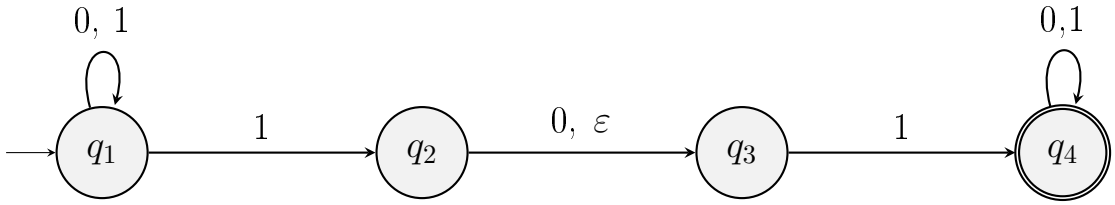
1. $L = \{w | w$ содержит четное число 0 и каждый 0 следует хотя бы за одной $1\}$;
2. $L = \{w | w$ начинается с 0 и содержит хотя бы одну $1\}$;
3. $L = \{w | w$ содержит нечетное число символов 0 и заканчивается $01\}$.

2 Недетерминированные конечные автоматы

Определение 2.1. Недетерминированным конечным автоматом (НКА) называется пятерка объектов $N = (Q, \Sigma, \delta, q_0, F)$, где

1. Q – конечное множество состояний;
2. Σ – конечное множество символов (алфавит);
3. $\delta : Q \times \Sigma_\varepsilon \to P(Q)$ – функция перехода, $P(Q)$ – множество подмножеств множества Q, $\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$;
4. $q_0 \in Q$ – начальное состояние;
5. $F \subset Q$ – множество финальных (допустимых) состояний.

Пример 2.1. Пример НКА



В НКА из одного состояния может быть несколько переходов, помеченных одним и тем же символом алфавита (или же используются не все символы алфавита). Введем также символ пустого слова ε.

Обработка слова $w = 01011$:

0: $\to q_1$

1: $q_1 \to q_1, \quad q_1 \to q_2, \quad q_1 \to (\varepsilon)q_3$

0: $q_1 \to q_1, \quad q_2 \to q_3, \quad \cancel{\to q_3}$

1: $q_1 \to q_1, \quad q_1 \to q_2 \quad q_1 \to q_3 \qquad q_3 \to q_4$

1: $q_1 \to q_1, \quad q_1 \to q_2, \quad q_1 \to q_3 \qquad \cancel{q_4} \qquad q_3 \to q_4, \; q_4 \to q_4$

Слово принимается автоматом (одно из состояний финальное).

Определение 2.2. Определение распознаваемости НКА. Пусть $N = (Q, \Sigma, \delta, q_0, F)$ – НКА и $w = w_1 w_2 \ldots, w_n \in \Sigma^*$ – строка длины n над Σ_ε, то есть $w_i \in \Sigma$ или $w_i = \varepsilon$. Говорят, что автомат N принимает слово w, если существует последовательность состояний $r_0, r_1, r_2, \ldots, r_n \in Q$, для которых выполнено: (1) $r_0 = q_0$; (2) $r_{i+1} \in \delta(r_i, w_{i+1})$, $i = 0, 1, \ldots, n-1$; (3) $r_n \in F$.

Задачи

Все языки рассматриваются над алфавитом $\Sigma = \{0, 1\}$.

2.1. Постройте НКА с фиксированным числом состояний.

1. $L = \{w \mid w$ заканчивается $00\}$, три состояния;
2. $L = \{w \mid w$ содержит $0101\}$, пять состояний;
3. $L = \{w \mid w$ содержит четное число 0 или ровно две $1\}$, шесть состояний.

2.2. Постройте НКА с тремя состояниями для языка $L = \{w \mid w$ заканчивается $00\}$. Преобразуйте НКА в ДКА.

2.3. Постройте НКА с тремя состояниями для языка $L = \{0^*1^*0^+\}$. Преобразуйте НКА в ДКА.

Здесь используется обозначение $R^+ = RR^*$, где R^* – операция Клини. Например, запись $0^*1^*0^+$ обозначает слова, у которых вначале стоят нули, далее единицы, в конце слова стоит не меньше одного нуля.

2.4. Постройте НКА для $A \cup B$.

1. $A = \{w \mid w = 1w'0\}$, $B = \{w \mid w = w'111\}$
2. $A = \{w \mid w = w'0101w''\}$, $B = \{w \mid w[2] = 0\}$ (третий символ равен 0, длина больше или равна трем).

2.5. Постройте НКА с фиксированным числом состояний.

1. Одно состояние. $L = \{\varepsilon\}$;
2. Одно состояние. $L = \{0^*\}$.

2.6. Постройте НКА с фиксированным числом состояний. Преобразуйте НКА в ДКА.

1. $L = \{0\}$, два состояния;
2. $L = \{1^*(001^+)^*\}$, три состояния (язык состоит из слов, у которых после каждых двух нулей следует как минимум одна единица).

2.7. Каждый из языков упражнения является пересечением или объединением более простых языков. Постройте ДКА для каждой простой части, а затем – автомат для исходного языка.

1. $\{w|w$ содержит четное число букв 0 и каждая 0 следует хотя бы за одной $1\}$;
2. $\{w|w$ начинается с 0 и содержит хотя бы одну $1\}$;
3. $\{w|w$ содержит нечетное число букв 0 или заканчивается $01\}$.

3 Регулярные операции и лемма о разрастании

Теорема 3.1. Если A, B – регулярные языки, то $A \cup B$ – регулярный язык.

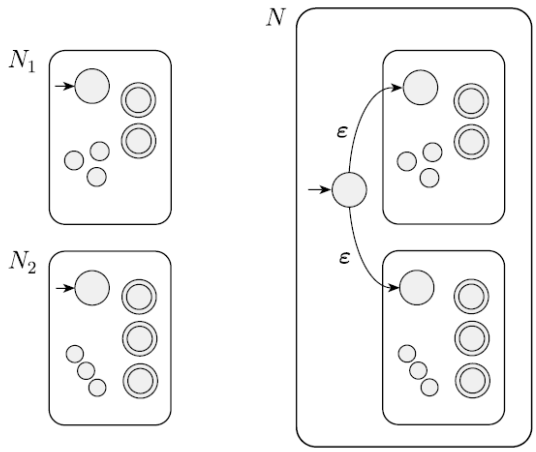


Рис. 1: Объединение языков.

Теорема 3.2. Если A – регулярный язык, то A^* – регулярный язык.

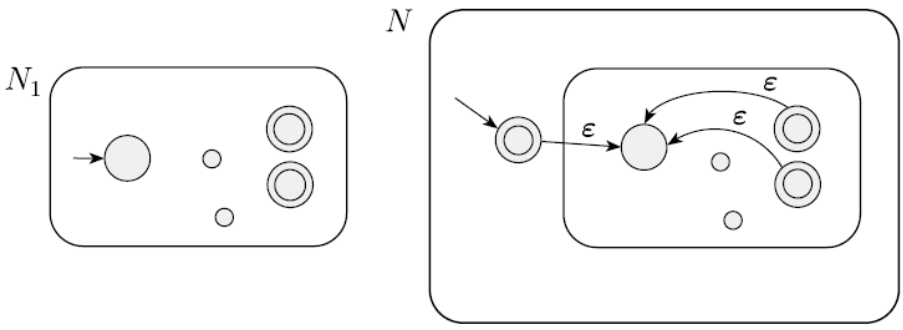


Рис. 2: Операция Клини.

Теорема 3.3. Если A, B – регулярные языки, то $A \circ B$ – регулярный язык.

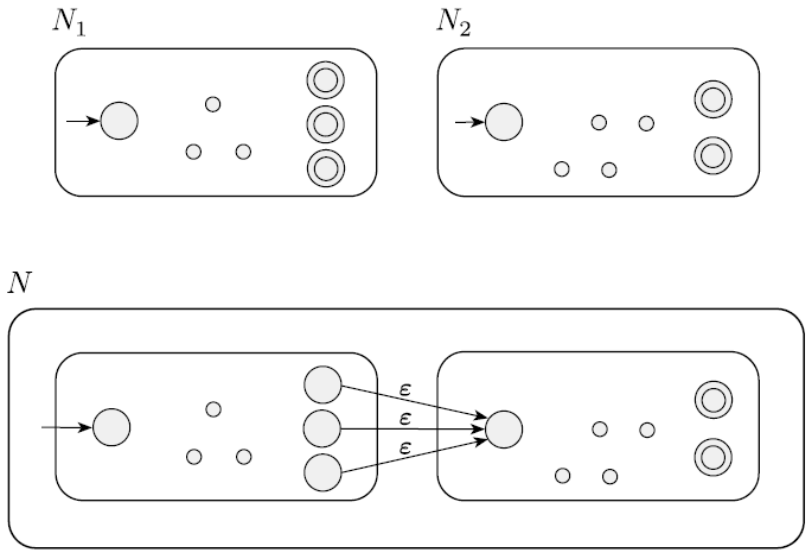


Рис. 3: Конкатенация языков.

Демма 3.1 (Лемма о разрастании, pumping lemma)**.** Пусть A – регулярный язык. Тогда существует такое число p, что любая строка s длины, большей чем p, может быть представлена в виде

$$s = xyz,$$

где x, y, z удовлетворяют условиям:

1. для каждого $i \geq 0$ выполнено $xy^iz \in A$;
2. длина $|y| > 0$;

3. $|xy| \leq p$.

Пример 3.1. Язык $B = \{0^n 1^n \mid n \geq 0\}$ нерегулярный.

Доказательство. Предположим противное. Пусть p – параметр из леммы о накачке. Рассмотрим $0^p 1^p = xyz$. По лемме, $|xy| < p$. Тогда y состоит только из 0. Но это приводит к противоречию, так как слово $xy^i z$ содержит больше нулей, чем единиц при $i > 0$.

Пример 3.2. Язык $C = \{w \mid w$ содержит одинаковое количество 0 и 1$\}$ нерегулярный.

Доказательство. Предположим противное. Если C – регулярный, то $C \cap 0^*1^*$ – регулярный (легко доказать, что второй язык регулярен). Но $C \cap 0^*1^* = B$ – противоречие.

Задачи

Все языки рассматриваются над алфавитом $\Sigma = \{0, 1\}$.

3.1. Постройте НКА, распознающий конкатенацию языков L_1 и L_2, где $L_1 = \{w \mid$ длина $w \leq 5\}$ и $L_2 = \{w \mid$ на всех нечетных позициях стоят 1$\}$.

3.2. Постройте НКА, распознающий язык A^*, где $A = \{w \mid w$ содержит не менее трех 1$\}$.

3.3. Использовав лемму о разрастании, покажите, что следующие языки не являются регулярными:

1. $L = \{ww \mid w \in \{0,1\}^*\}$ (указание: рассмотреть слово $0^p 10^p 1$);
2. $L = \{0^i 1^j \mid i > j\}$ (указание: $xy^0 z \in L(M)$).

3.4. Покажите, что любой НКА может быть приведен к автомату только с одним финальным состоянием.

3.5. Постройте НКА, распознающий конкатенацию языков L_1 и L_2.
$L_1 = \{w \mid w$ содержит не менее трех 1$\}$ и $L_2 = \emptyset$.

3.6. Постройте НКА, распознающий язык A^*, где $A = \{w \mid w$ содержит не менее двух 0 и не более одной 1$\}$.

3.7. Использовав лемму о разрастании, покажите, что следующие языки не являются регулярными:

1. $L = \{0^n 1^n 2^n \mid n \geq 0\}$;
2. $L = \{www \mid w \in \{0,1\}^*\}$.

4 Контекстно-свободные грамматики

Определение 4.1. Контекстно-свободной грамматикой (КС-грамматикой) называется четверка

$$G =< V, \Sigma, R, S \in N >,$$

где

1. Σ — множество терминалов (алфавит);
2. V – конечное множество переменных (нетерминалов);
3. R – конечное множество правил вывода вида, каждое правило имеет вид $A \to \gamma$, где $\gamma \in (\Sigma \cup V)^*$, а $A \in N$;
4. $S \in V$ – стартовый нетерминал.

Пример 4.1. Рассмотрим КС-грамматику G_1 с тремя правилами вывода:

$$A \to 0A1 \qquad (1)$$

$$A \to B \qquad (2)$$

$$B \to \# \qquad (3)$$

Алфавит $\Sigma = \{0, 1, \#\}$, $V = \{A, B\}$, A – стартовая переменная.

Пример слова, которые порождает грамматика:

$$A \Rightarrow_1 0A1 \Rightarrow_1 00A11 \Rightarrow_2 00B11 \Rightarrow_3 00\#11.$$

Аналогично можно получить любое слово вида $0^n\#1^n$. Если заменить символ $\#$ на ε (пустое слово), то получим все слова вида 0^n1^n.

Порождение слов при помощи грамматики происходит следующим образом:

1. Начинаем порождать слова со стартовой переменной по одному из правил в R;

2. Если в полученном выражении есть переменные, то заменяем их по одному из правил;

3. Если в выражении только терминальные символы, то останавливаемся.

Для удобства записи, если слева стоит одна и та же переменная, будем писать выражения справа через $|$.

Таким образом, правила предыдущего примера можно записать в виде

$$A \to 0A1 \mid B$$

$$B \to \#$$

Пример 4.2. Рассмотрим КС-грамматику G_2 с тремя правилами вывода:

$$S \to (S) \mid SS \mid \varepsilon$$

Примеры вывода слов, порождаемых грамматикой:

$$S \Rightarrow SS \Rightarrow SSS \Rightarrow S(S)S \Rightarrow S((S))S \Rightarrow$$

$$\Rightarrow (S)((S))S \Rightarrow (\varepsilon)((S))S \Rightarrow (\varepsilon)((\varepsilon))S \Rightarrow (\varepsilon)((\varepsilon))\varepsilon$$

$$S \Rightarrow (S) \Rightarrow ((S)) \Rightarrow ((SS)) \Rightarrow ((\varepsilon S)) \Rightarrow ((\varepsilon\varepsilon))$$

Здесь алфавитом является $\Sigma = \{(,)\}$, переменные $V = \{S\}$. Таким способом строится язык правильных скобочных последовательностей.

Задачи

4.1. Приведите по два примера строк, удовлетворяющих и не удовлетворяющих регулярному выражению (всего по четыре на каждое упражнение): 1) a^*b^*; 2) $a(ba)^*b$.

4.2. Постройте НКА, соответствующие регулярным выражениям:

1) $(01 \cup 001 \cup 010)^*$;

2) $(0 \cup 1)^*000(0 \cup 1)^*$.

Постройте контекстно-свободную грамматику, порождающую заданный язык над алфавитом $\Sigma = \{0, 1\}$. В этих упражнениях удобно сначала построить ДКА, а потом – с его помощью КС-грамматику.

4.3. $L = \{w \mid w$ не содержит подслов $110\}$.

4.4. $L = \{w \mid$ длина $w \leq 5\}$.

Постройте контекстно-свободную грамматику, порождающую заданный язык.

4.5. $L = \{a^i b^j \mid i \geq j\}$.

4.6. $L = \{a^i b^j \mid i < j\}$.

4.7. $L = \{a^i b^i c^k \mid i, k \geq 0\}$.

4.8. $L = \{a^i b^k c^k \mid i, k \geq 1\}$.

4.9. Приведите по два примера строк, удовлетворяющих и не удовлетворяющих регулярному выражению (всего по четыре на каждое упражнение): 1) $a^* \cup b^*$; 2) $\Sigma^* a \Sigma^* b \Sigma^* a \Sigma^*$; 3) $(aaa)^*$.

4.10. Постройте НКА, соответствующие регулярным выражениям: 1) $(((00)^*(11)) \cup 01)^*$; 2) $\emptyset^*$.

Постройте контекстно-свободную грамматику, порождающую заданный язык над алфавитом $\Sigma = \{0, 1\}$. В этих упражнениях удобно сначала построить ДКА, а потом с его помощью – КС-грамматику.

4.11. $L = \{w \mid$ все слова, кроме $11, 111\}$.

4.12. $L = \{w \mid w$ не содержит строки $01\}$

Постройте контекстно-свободную грамматику, порождающую заданный язык.

4.13. $L = \{a^i b^j c^k \mid i, j, k \geq 1, i = j$ или $j = k\}$.

4.14. $L = \{ww^R \mid w \in \{a, b\}^*\}$, где w^R – обращение слова w.

4.15. $L = \{w \mid w = w^R, w \in \{a, b\}^*\}$, где w^R – обращение слова w.

5 Машины Тьюринга

Одноленточная машина Тьюринга работает с неограниченной в обе стороны лентой, разбитой на ячейки. В каждой ячейке записана одна буква рабочего (ленточного) алфавита Σ. Предполагается, что есть специальный символ "пробел" (например, "B") для обозначения пустых ячеек. Имеется устройство, которое может читать содержимое ячейки и записывать в нее символы алфавита, а также перемещаться вдоль ленты. Устройство может находиться в одном из конечного множества состояний Q.

Устройство исполняет программу, состоящую из команд вида

$$qa \to q'a'S,$$

где q – состояние, в котором находится устройство, a – символ, который находится в читаемой ячейке, q' – состояние, в которое переходит устройство, a' – символ, который заменяет символ a, S – это один из символов $R, L, -$, указывающий, в какую сторону сдвигается устройство.

В программе содержится ровно по одной команде с каждой возможной левой частью (qa), порядок несущественен. В каждый момент времени исполняется ровно одна команда с подходящей левой частью. В множестве всех состояний Q выделены начальное состояние, с которого устройство начинает работу, и заключительное состояние, в котором работа прекращается.

Соглашения:

1. Начальное состояние – q_0, заключительные — q_{ac}, q_{rej}.
2. Натуральные числа представляются на ленте в унарной записи – число n записывается как $11\ldots1$.
3. В начале работы конечный кусок ленты заполнен входными данными, а все остальные клетки содержат символ B. При этом устройство расположено непосредственно слева от входных данных.
4. После завершения работы результатом является то слово на ленте (последовательность букв между соседними "B"), на которое указывает устройство. Последнее означает, что устройство остановилась внутри слова или непосредственно слева от него. В частности, если устройство остановилась на букве B, и в соседней справа клетке также стоит B, то результат – пустое слово.
5. При вычислении функций нескольких переменных аргументы на ленте разделяются одним символом B.
6. При вычислении частичной функции машина Тьюринга останавливается в том и только том случае, когда функция определена.

Пример 5.1. Заменить во входном слове из 0 и 1 все буквы 0 на 1 и наоборот.

Решение.

$q_0B \to q_1BR$
$q_10 \to q_11R$
$q_11 \to q_10R$
$q_1B \to q_{acc}BL$

Задачи

Написать программы для машины Тьюринга, выполняющие следующие преобразования слов:

5.1. Переместить 0 через блок единиц ($B011\ldots1B \to B11\ldots10B$).

5.2. Во входном слове из 0 и 1 переместить первую букву в конец слова.

5.3. Удвоение блока из 1 (используйте дополнительную букву x в качестве двойника 1).

5.4. Обратить слово из 0 и 1 (на выходе буквы слова в обратном порядке).

5.5. Прибавить 1 к натуральному числу, записанному в двоичной записи.

5.6. Преобразовать унарную запись натурального числа в его двоичную запись.

Ниже предполагается, что натуральные числа записываются на ленте машины Тьюринга в унарной записи, несколько аргументов разделяются пробелами (B). Написать программы для машины Тьюринга, вычисляющие следующие функции натурального аргумента:

5.7. $f(x, y) = x + y$.

5.8. $f(x) = x - 1$ для $x > 0$ и $f(x) = 0$, если $x = 0$.

5.9. $f(x) = 0$, если x четно, и $f(x) = 1$, если x нечетно.

5.10. $f(x, y) =\mid x - y \mid$. Указание: стирать по одной единице из записей x и y, пока один из этих блоков единиц не станет пустым. Учесть, что последняя стертая единица — лишняя, и ее надо восстановить.

5.11. $f(x) = 2x$.

5.12. $f(x) = x \mod 2$.

6 Пример проверочной работы-1

Пример первой проверочной работы.

1. Постройте ДКА, распознающий язык

$$L = \{w \mid \text{ все слова из } \{0,1\}^+ \text{ кроме, } 00, 101\}.$$

2. Постройте НКА, распознающий язык $L = \{0^*1^*0^+\}$, три состояния.

3. Преобразуйте НКА из второго задания в ДКА.

4. Постройте КС-грамматику, порождающую язык

$$L = \{w \mid w \in \{0,1\}^* \text{ не содержит строки } 01\}.$$

5. Постройте машину Тьюринга, которая вычисляет $f(x) = x \mod 2$. Натуральные числа записывать в унарной записи.

7 Арифметика остатков

Теорема 7.1 (Деление с остатком)**.** Если a и b – целые числа и $b \neq 0$, то существует единственная пара чисел q и r таких, что $a = bq + r$, $0 \leq r < |b|$.

В этом случае говорят, что a делится на b с остатком.

Пример 7.1.

$$\mathbf{11} = \mathbf{3} \cdot 3 + 2$$

$$-\mathbf{11} = \mathbf{3} \cdot (-4) + 1$$

$$\mathbf{11} = -\mathbf{3} \cdot 4 + 1$$

Определение 7.1. Число r в указанном представлении называется **остатком** от деления a на b. Если остаток равен нулю, то говорят, что a **делится** на b.

Теорема 7.2. Если два числа a и b делятся на третье число c, то их сумма $a + b$ и разность $a - b$ делятся на это c. Для делимости произведения достаточно делимости одного из сомножителей: если a делится на c, то и ab делится на c, каково бы ни было (целое) b.

Определение 7.2. Если два числа a и b дают одинаковые остатки при делении на положительное число n, то говорят, что они **сравнимы по модулю** n, и пишут

$$a = b \mod n.$$

Эквивалентное определение: a и b сравнимы по модулю n, если разность $a - b$ делится на n.

Таблицы сложения и умножения по модулю n

Случай $n = 5$.

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3				
4	4				

*	1	2	3	4
1	1	2	3	4
2	2	4	1	2
3	3			
4	4			

Наибольший общий делитель. Алгоритм Евклида

Определение 7.3. Пусть a, b – целые числа. Целое число d называется делителем a и b, если оно делит оба числа (без остатка).

Определение 7.4. Целое число d называется наибольшим общим делителем (НОД) целых чисел a и b, если

1) d является общим делителем этих чисел;

2) d делится на любой общий делитель чисел a, b.

Наибольший общий делитель обозначают НОД(a,b) или $\gcd(a, b)$.

Алгоритм Евклида нахождения НОД.

$r_0 = a$

$r_1 = b$

$r_2 = r_0 - r_1 q_1,\ 0 \le r_2 < |r_1|$

$\dots$

$r_{k+1} = r_{k-1} - r_k q_k,\ 0 \le r_{k+1} < |r_k|$

Алгоритм продолжает работу до тех пор, пока r_{k+1} не станет равным нулю. Наибольший общий делитель

$$\text{НОД}(a, b) = r_k.$$

Расширенный алгоритм Евклида

$r_0 = a,\ s_0 = 1,\ t_0 = 0$

$r_1 = b,\ s_1 = 0,\ t_1 = 1$

$r_2 = r_0 - r_1 q_1,\ 0 \le r_2 < |r_1|,\ s_2 = s_0 - q_1 s_1,\ t_2 = t_0 - q_1 t_1$

$\dots$

$r_{k+1} = r_{k-1} - r_k q_k,\ 0 \le r_{k+1} < |r_k|,\ s_{k+1} = s_{k-1} - q_k s_k,\ t_{k+1} = t_{k-1} - q_k t_k$

Алгоритм продолжает работу до тех пор, пока r_{k+1} не станет равным нулю. Наибольший общий делитель

$$r_k = (a, b) = as_k + bt_k.$$

Расширенный алгоритм Евклида позволяет находить (мультипликативный) обратный по модулю числа.

Задачи

7.1. Найдите частное и остаток при делении с остатком числа a на b:

a) $a = 19,\ b = 7$; b) $a = -111,\ b = 11$;

c) $a = 789,\ b = 23$; d) $a = 1001,\ b = 13$.

e) $a = 17,\ b = 5$; f) $a = 17,\ b = -5$.

7.2. Докажите, что если $a \equiv 1 \mod n$ и $b \equiv 1 \mod n$, то $ab \equiv 1 \mod n$.

7.3. Докажите, что остаток при делении квадрата нечетного натурального числа на 8 равен 1.

7.4. Докажите, что $5 | m^5 - m$.

7.5. Докажите, что $6 | m(m+1)(2m+1)$.

7.6. Докажите, что

a) $a^{10} - 9a + 8$ делится на 2,

b) $a^5 + 3a^3 - 12$ делится на 4,

c) $a^3 - 7a + 18$ делится на 6,

7.7. Найдите наибольший общий делитель, использовав алгоритм Евклида

a) $\gcd(1, 5)$; b) $\gcd(100, 101)$; c) $\gcd(123, 277)$;

d) $\gcd(1529, 14039)$; e) $\gcd(1529, 14038)$; f) $\gcd(11111, 111111)$.

7.8. Найдите наибольший общий делитель и его представление в виде $(a, b) = au + bv$, использовав расширенный алгоритм Евклида

a) $\gcd(12, 18)$; b) $\gcd(111, 201)$; c) $\gcd(1001, 1331)$.

7.9. Найдите частное и остаток при делении с остатком числа a на b:

a) $a = 0,\ b = 19$; b) $a = 3,\ b = 5$;

c) $a = -1,\ b = 3$; d) $a = 4,\ b = 1$.

7.10. Докажите, что сумма квадратов двух последовательных натуральных чисел при делении на 4 дает остаток 1.

7.11. Докажите, что $6|m^3 + 5m$.

7.12. Докажите, что $9|4^n + 15n - 1$ для любых положительных целых n.

7.13. Докажите, что

a) $a^7 - a - 56$ делится на 7,

b) $a^5 - 17a^3 + 24$ делится на 8,

c) $a^9 + 17a^3 - 18$ делится на 9.

7.14. Найдите наибольший общий делитель и его представление в виде $(a, b) = au + bv$, использовав расширенный алгоритм Евклида

a) $\gcd(12345, 54321)$; b) $\gcd(1000, 5040)$; c) $\gcd(9888, 6060)$.

8 Криптосистема RSA

Первым примером реализации криптосистемы с отрытым ключом стала система RSA (аббревиатура от фамилий Rivest, Shamir и Adleman), основанная на вычислительной сложности задачи разложения числа на простые сомножители. Криптосистема RSA стала первой системой, пригодной и для шифрования, и для цифровой подписи.

Формирование открытого и секретного ключа

Алиса подбирает два больших простых числа p и q. Держа их в секрете, Алиса публикует

$$n = pq.$$

Кроме того, Алиса выбирает шифрующую компоненту e, удовлетворяющую условию

$$\text{НОД}(e, (p-1)(q-1)) = 1.$$

Пара (n, e) составляет **открытый ключ** Алисы.

Для выбора секретного ключа Алиса применяет расширенный алгоритм Евклида к паре $e, (p-1)(q-1)$, получая при этом расшифровывающую экспоненту d, такую, что

$$ed = 1 \mod (p-1)(q-1).$$

d – **секретный ключ** Алисы.

Шифрование/Расшифрование

Боб хочет послать сообщение m Алисе. Сообщение – это некоторое число $1 \leq m \leq n$. Шифротекст, которое Боб формирует на основе открытого ключа:

$$C = m^e \mod n.$$

Алиса, получив сообщение C, дешифрует его по правилу

$$D(C) = C^d \mod n.$$

Теорема 8.1.

$$(M^e)^d = M \mod n.$$

Пример 8.1. Пусть $p = 5$, $p = 11$. Тогда $n = 5 \cdot 11 = 55$ и $\varphi(n) = (p-1)(p-1) = 4 * 10 = 40$.

Открытые ключи $n = 55$, $e = 7$. Секретный ключ $d = 23$.

Шифровка сообщения $m = 6$:

$$C = m^7 = 6^7 \mod 55 = 63636 \mod 55 = 41 \mod 55.$$

Дешифровка сообщения:

$$C^{23} = 41^{23} \mod 55 = \ldots = 6 \mod 55.$$

Алгоритмы цифровой подписи RSA

Криптосистема RSA позволяет реализовать цифровую подпись. Рассмотрим самую простую версию такой подписи.

Пусть Алиса хочет подписать документ m. Для этого она использует свой закрытый ключ d. Для создания подписи, обозначаемой s, Алиса вычисляет

$$s = m^d \text{ н.о.д. } n.$$

Пара (s, m) состоит из подписи и открытого текста.

Для проверки неизменности сообщения от Алисы с помощью электронной подписи Боб использует подпись и открытый ключ Алисы. Вычислив

$$m' = s^E \mod n,$$

Боб может убедится, что сообщение послано Алисой, и текст не изменялся.

Задачи

8.1. Для передачи секретной информации выбрана криптосистема RSA. Использовав открытый ключ (n, e), нужно передать Алисе секретное сообщение m и дешифровать его с помощью секретного ключа. Даны следующие параметры криптосистемы: $p = 17, q = 31, e = 7, m = 2$.

8.2. Алиса и Боб используют различные системы RSA с общим модулем n и публичными экспонентами шифрования e_A и e_B (держат в секрете свои экспоненты дешифрования d_A и d_B). Докажите, что Алиса может дешифровывать сообщения, посланные Бобу. Кроме того, покажите, что криптоаналитик Ева может дешифровывать сообщения, посланные Алисе и Бобу, если $(e_A; e_B) = 1$.

8.3. Зашифруйте сообщение ATTACK, использовав криптосхему RSA с $n = 43 \cdot 59$ и $e = 13$. Каждая буква преобразуется в ее номер в алфавите (A-00, B-01, ..., Z-25).

8.4. Дешифруйте сообщение 0667 1947 0671, зашифрованное криптосхемой RSA с $n = 43 \cdot 59$ и $e = 13$.

8.5. Для электронной подписи используется криптосистема RSA. Секретный ключ Боба составляют числа p_B и q_B, а секретный ключ Алисы – числа p_A и q_A. Пусть открытыми ключами Боба и Алисы являются пары $n_B = \{p_B q_B; eB\}$ и $n_A = \{p_A q_A; e_A\}$ соответственно. Необходимо передать Бобу секретное поручение m от Алисы, а также удостовериться в подлинности данного сообщения. Параметры криптосистемы: $p_A = 11, q_A = 23, e_A = 31, p_B = 7, q_B = 13, e_B = 5, m = 41$.

8.6. Для передачи секретной информации выбрана криптосистема RSA. Использовав открытый ключ (n, e), нужно передать Алисе секретное сообщение m и дешифровать его с помощью секретного ключа. Даны следующие параметры криптосистемы: $p = 11, q = 17, e = 9, m = 3$.

8.7. Дешифруйте сообщение 185 2038 2460 2550, зашифрованное криптосхемой RSA с $n = 53 \cdot 61$ и $e = 17$.

8.8. Докажите, что в система RSA с модулем 21 и 35 все возможные ключи шифрования e совпадают с ключами дешифрования d.

8.9. Для электронной подписи используется криптосистема RSA. Секретный ключ Боба составляют числа p_B и q_B, а секретный ключ Алисы – числа p_A и q_A. Пусть открытыми ключами Боба и Алисы являются пары $n_B = \{p_B q_B; eB\}$ и $n_A = \{p_A q_A; e_A\}$ соответственно. Необходимо передать Бобу секретное поручение m от Алисы, а также удостовериться в его подлинности. Параметры криптосистемы: $p_A = 7, q_A = 11, e_A = 7, p_B = 3, q_B = 5, e_B = 7, m = 13$.

9 Алгебраические структуры. Группы

Определение 9.1. Множество с бинарной операцией $(G, \circ)$ называется *группой*, если выполнены три условия (аксиомы группы):

1. Операция *ассоциативна*, т. е.

$$a \circ (b \circ c) = (a \circ b) \circ c \quad \text{для всех } a, b, c \in G.$$

2. Есть *нейтральный элемент*, т. е. такой элемент $e \in S$, что $e \circ a = a \circ e = a$ для любого $a \in G$.

3. Для каждого элемента $a \in G$ найдется *обратный элемент*, т. е. такой $b \in G$, что $a \circ b = b \circ a = e$.

Обратный элемент обозначается a^{-1}. Группу принято обозначать $(G, \circ)$ или просто G, когда понятно, о какой операции идет речь. Обычно символ $\circ$ для обозначения операции опускают и пишут просто ab.

Определение 9.2. Группа G называется *коммутативной* или *абелевой*, если групповая операция *коммутативна*, т. е. $ab = ba$ для любых $a, b \in G$.

Если в случае произвольной группы G принято использовать мультипликативные обозначения для групповой операции — gh, e, g^{-1}, то в теории абелевых групп чаще используют аддитивные обозначения, т. е. $a+b$, 0, $-a$.

Определение 9.3. *Порядок* группы G – это число элементов в G. Группа называется *конечной*, если её порядок конечен, и *бесконечной* иначе. Порядок группы G обозначается $|G|$.

Примеры групп.

1) Числовые аддитивные группы: $(\mathbb{Z}, +)$, $(\mathbb{Z}_n, +)$.

2) Числовые мультипликативные группы: $\mathbb{Q}^* = \mathbb{Q} \setminus \{0\}$, $\mathbb{R}^* = \mathbb{R} \setminus \{0\}$.

3) Группы матриц: $GL_n(\mathbb{R}) = \{A \in Mat(n \times n, \mathbb{R}) \mid \det(A) \neq 0\}$.

4) Группы подстановок: симметрическая группа S_n — все подстановки длины n, $|S_n| = n!$

Определение 9.4. Пусть G – группа и $g \in G$. *Порядком* элемента g называется такое наименьшее натуральное число m, что $g^m = e$. Если такого натурального числа m не существует, говорят, что порядок элемента g равен бесконечности.

Порядок элемента обозначается $ord(g)$. Заметим, что $ord(g) = 1$ тогда и только тогда, когда $g = e$.

Задачи

9.1. Докажите, что обратный элемент единственный.

9.2. Найдите произведение перестановок $\tau \cdot \sigma$

$$\tau = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 5 & 4 & 1 & 7 & 3 & 6 & 2 \end{pmatrix}; \qquad \sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 4 & 2 & 1 & 7 & 3 & 5 & 6 \end{pmatrix}$$

9.3. Найдите порядок элемента $\tau = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 5 & 4 & 1 & 7 & 3 & 6 & 2 \end{pmatrix}$.

9.4. Выпишите все элементы группы S_3. Найдите их порядки.

9.5. Рассмотрим матрицы

$$a = \begin{pmatrix} \cos(2\pi/4) & -\sin(2\pi/4) \\ \sin(2\pi/4) & \cos(2\pi/4) \end{pmatrix}, \qquad b = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

Докажите, что: a) порядок элемента a равен четырем, а порядок b равен двум; b) $bab = a^{-1}$; c) покажите, что множество $1, a, a^2, a^3, b, ba, ba^2, ba^3$ образует группу (это группа симметрий правильного 4-угольника).

9.6. Найдите порядки всех элементов в $(\mathbf{Z}_6, +)$.

9.7. Найдите порядки всех элементов в $(\mathbf{Z}_6^*, \times)$.

9.8. Рассмотрим эллиптическую кривую (множество точек, удовлетворяющих уравнению) $y^2 = x^3 + 3x + 2$ над полем $\mathbb{Z}_5$. На этой кривой, в частности, лежит точка $(1, 1)$, так как $1^2 \equiv 1^3 + 3 \cdot 1 + 2 \pmod 5$. Нужно проверить, что $(1, \pm 1)$, $(1, \pm 4)$, $(2, \pm 1)$, $(2, \pm 4)$ – это все точки данной кривой.

Пусть $E_p(a, b)$ – множество точек на эллиптической кривой, заданной уравнением

$$y^2 = x^3 + ax + b \mod p,$$

где $p \neq 2, 3$.

Добавим к множеству точку O, координаты которой приято обозначать (x, ∞). Введем операцию сложения на множестве точек:

1) $O + O = O$, $P(x, y) + O = P(x, y)$, $P(x, y) + P(x, -y) = O$,

2) Если $P(x_1, y_1) \neq \pm Q(x_2, y_2)$, то

$$R(x, y) = P + Q = (\lambda^2 - x_1 - x_2, \lambda(x_1 - x_3) - y_1),$$

где $\lambda = (y_2 - y_1)/(x_2 - x_1)$.

3) Если $P(x_1, y_1) = Q(x_2, y_2)$, то

$$R(x, y) = P + P = 2P = (\lambda^2 - x_1 - x_2, \lambda(x_1 - x_3) - y_1),$$

где $\lambda = (3x_1^2 + a)/(2y_1)$.

Вычислите по формулам $(1, 1) + (1, 4)$ и $2(1, 1)$.

9.9. Найдите все точки эллиптической кривой $E_7(2, 6)$. Постройте таблицу умножения. Найдите порядки элементов.

10 Блоковые коды, исправляющие ошибки. Линейные коды

Мы рассматриваем передачу сообщений блоками размера n над некоторым алфавитом X.

Общая схема работы подразумевается следующей:

- Источник формирует сообщение $u = (u_1, u_2, \ldots, u_k)$ длины k.
- Кодер добавляет проверочные символы к сообщению (само сообщение тоже может изменятся).

- Сообщение передается по каналу связи, в котором возможны помехи.
- Декодер проверяет наличие ошибок, при возможности исправляет их и передает декодированное сообщение получателю.

Пример 10.1. Код с повторениями. Пользователь хочет передать сообщение 1 или 0 (информационное сообщение, $k = 1$). Для уменьшения ошибки символ дублируется 5 раз. По каналу связи будет передано либо 00000, либо 11111.

Если будет получено, например, сообщение 01000, то, значит, в момент передачи произошла ошибка (обнаружение ошибки).

Если вероятность ошибки в каждом символе меньше 0.5, то вероятнее всего было передано сообщение 00000 (исправление ошибки).

Пример 10.2. Код проверки на четность. Пользователь передает сообщение $(u_1, u_2, \ldots, u_k)$, состоящее из 0 и 1. Кодер добавляет один дополнительный символ $u_{k+1} \in \{0, 1\}$ так, чтобы сумма $u_1 + u_2 \ldots + u_k + u_{k+1}$ была четной. Если в канале произойдет одна ошибка (или любое нечетное количество ошибок), то сумма станет нечетной, и мы зафиксируем наличие ошибки.

В этом случае восстановить сообщение мы не сможем. Такие коды используют в системах, где вероятность ошибки очень маленькая, например, в вычислительных машинах для контроля передач информации между регистрами и контроля считываемой информации в оперативной памяти.

В качестве алфавита берут обычно какой-либо алгебраический объект, в приложениях это чаще всего конечное поле. Есть исследования, в которых алфавит – это конечные кольца, группы и полугруппы, лупы.

Определение 10.1. Кодом длины n называется любое непустое подмножество $C \in F^n$. Если количество элементов $|C| = M$, то говорят, что C является (n, M)-кодом.

Элементы кода называются кодовыми словами.

Величина $k = \log_q M$ называется размерностью (или информационной длиной) кода C. Информационная длина показывает долю полезных (информационных) символов, которые передаются по каналу связи.

Фактически декодер проверяет, принадлежит ли принятое слово множеству C или нет. Если слово не принадлежит C, значит, произошла ошибка. Далее будем решать следующие задачи:

1. Как задать C, чтобы проверка принадлежности была простой.

2. Если произошла ошибка, то можно ли восстановить исходное сообщение.

3. Как увеличить долю информационных символов, сохранив корректирующие возможности кода.

Определение 10.2. *Расстояние Хэмминга* на множестве F^n определяется следующим образом:

$$d(x, y) = |\{i | x_i \neq y_i\}|,$$

где $x = (x_1, \ldots, x_n),\ y = (y_1, \ldots, y_n),\ x, y \in X^n$.

Теорема 10.1. Расстояние Хэмминга задает на F^n структуру метрического пространства.

Определение 10.3. *Минимальное расстояние (кодовое расстояние)* кода определяется следующим образом:

$$d(C) = \min d(x, y),$$

где минимум берется по всем парам элементов $x, y \in C,\ x \neq y$.

Теорема 10.2. Если кодовое расстояние $d(C) = 2t+1$, то код может исправлять t и обнаруживать $2t$ ошибок.

Если кодовое расстояние $d(C) = 2t$, то код может исправлять $t-1$ и обнаруживать $2t-1$ ошибку.

Линейные коды

Для удобства кодирования и декодирования обычно рассматриваются коды с какой-либо алгебраической структурой. Наиболее важными с практической точки зрения являются линейные коды.

Далее в качестве алфавита берется конечное поле $F = GF(q)$, состоящее из q элементов.

Определение 10.4. *Линейным кодом* C называется подпространство векторного пространства F^n над полем F.

Размерность пространства C называется размерностью кода. Если k – размерность линейного пространства C и d – минимальное расстояние кода C, то говорят, что C линейный $[n,k,d]$-код над C.

Применяют также запись $[n,k]$-код и $[n,k,d]_q$-код для указания количества элементов в поле F.

Из определения ясно, что в C ровно q^k элементов.

Определение 10.5. *Весом Хэмминга* $wt(x)$ слова (вектора) $x \in F^n$ называется число ненулевых компонент x.

Теорема 10.3. Расстояние Хэмминга для слов выражается через вес Хэмминга следующим образом:

$$d(x,y) = wt(x-y).$$

Если код C линеен, то

$$d(C) = \min_{x \in C, x \neq 0} wt(x).$$

Линейный код как векторное подпространство можно задать либо через систему порождающих векторов (с помощью порождающей матрицы), либо как решение систем линейных уравнений (с помощью проверочной матрицы).

Мы можем задать (построить) линейное подпространство с помощью порождающей матрицы

$$C = \{uG \mid u = (u_1, u_2, \ldots, u_k),\ G - k \times n \text{ матрица}\}.$$

Строки порождающей матрицы образуют базис подпространства C.

Пример. Код проверки на четность

$$\{(u_1, u_2, \ldots, u_k, u_1 + u_2 + \ldots + u_k) = (u_1, u_2, \ldots, u_k)(E_k|A)\}.$$

$$(E_k|A) = \begin{pmatrix} 1 & 0 & \ldots & 0 & 1 \\ 0 & 1 & \ldots & 0 & 1 \\ . & . & . & \ldots & . \\ 0 & 0 & \ldots & 1 & 1 \end{pmatrix}.$$

Удобно представить матрицу G в виде $(E_k \mid A)$. В этом случае исходное сообщение $u = (u_1, u_2, \ldots, u_k)$ кодируется вектором (u, uA).

Для однозначного кодирования важно, чтобы из условия $u \neq u'$ всегда следовало, что

$$uG \neq u'G.$$

Это эквивалентно тому, что строки матрицы G линейно независимы. Таким образом, строки матрицы G образуют базис линейного пространства C.

Линейный код можно задать и как решение систем однородных линейных уравнений

$$C = \{x \in K^n \mid xH^t = 0,\ H - (n-k) \times n \text{ матрица}\}.$$

Определение 10.6. Матрица H называется *проверочной матрицей* кода C, если выполнено условие: $x \in C$ тогда и только тогда, когда

$$xH^t = 0.$$

Проверочная и порождающая матрицы связаны соотношением

$$GH^t = 0.$$

Если $G = (-A^t|E_k)$, то

$$H = (E_{n-k}|A).$$

Теорема 10.4. Если любые $s \leq d-1$ столбцов проверочной матрицы H линейного (n,k)-кода линейно независимы, то минимальное расстояние кода равно по меньшей мере d. Если при этом найдутся d линейно зависимых столбцов, то минимальное расстояние кода в точности равно d.

Теорема 10.5. Если минимальное расстояние линейного (n,k)-кода равно d, то любые $l \leq d-1$ столбцов проверочной матрицы H линейно независимы, и найдутся d линейно зависимых столбцов.

Задачи

В качестве алфавита рассматривается множество $F = GF(2) = \{0,1\}$.
10.1. Вычислите $d(11001, 01110)$.
10.2. Для данного множества $C = \{101010, 010110, 000001\}$ найдите (минимальное) кодовое расстояние и число ошибок, которые код обнаруживает и исправляет.

10.3. Пусть $C = \{000000, 100110, 010101, 001011, 101101, 011110, 110011, 111000\}$. Учитывая, что это $[6, 8, 3]_2$ код, восстановите сообщения 000001.

10.4. Найдите все слова, которые находятся на расстоянии 3 от слова 1010 в F^4.

10.5. Пусть $C = \{000000, 100110, 010101, 001011, 101101, 011110, 110011, 111000\}$. Зная, что это линейный код, найдите минимальное кодовое расстояние. Можете ли вы доказать, что это действительно линейный код? Найдите базисные векторы.

10.6. Вычислите $d(0000, 0110)$.

10.7. Для данного множества $C = \{01101010, 11000110, 00011001, 10101100\}$ найти (минимальное) кодовое расстояние и число ошибок, которые код обнаруживает и исправляет.

10.8. Найдите все слова, которые находятся на расстоянии 3 от слова 10101 в F^5.

10.9. Для данного множества $C = \{111100, 110011, 001111\}$ Найдите (минимальное) кодовое расстояние и число ошибок, которые код обнаруживает и исправляет.

10.10. Пусть $C = \{000000, 100110, 010101, 001011, 101101, 011110, 110011, 111000\}$. Учитывая, что это $[6, 8, 3]_2$ код, восстановите сообщения 011110.

10.11. Найдите проверочную и порождающую матрицы линейного кода

$$C = \{(x_1, x_2, x_3, x_4) \mid x_1 + x_2 + x_3 + x_4 = 0\}.$$

10.12. Пусть C_1 и C_2 – линейные коды одинаковой длины n с порождающими матрицами G_1 и G_2 и размерностями $k_1 \geq k_2 > 0$. Определим следующие (не обязательно линейные) коды: $C_3 = C_1 \cup C_2$; $C_4 = C_1 \cap C_2$; $C_5 = C_1 + C_2 = \{x + y \mid x \in C_1, y \in C_2\}$; $C_6 = \{(x, y) \mid x \in C_1, y \in C_2\} \subset F^{2n}$.

1. Покажите, что коды C_4, C_5, C_6 – линейные.
2. При каких условиях код C_3 является линейным?
3. Докажите, что кодовое расстояние $d(C_4) \geq \max(d(C_1), d(C_2))$.
4. Выразите порождающую матрицу кода C_6 через матрицы G_1 и G_2.
5. Докажите, что кодовое расстояние $d(C_6) = \min(d(C_1), d(C_2))$.

11 Декодирование линейного кода

Для линейных кодов проверка на отсутствие ошибок сводится к умножению на проверочную матрицу.

Исправление ошибок осуществляется следующим образом. Для принятого вектора x вычисляется

$$xH^t.$$

Пусть $e = (e_1, e_2, \ldots, e_n)$ – вектор ошибок, $c = (c_1, c_2, \ldots, c_n)$ – сообщение, которое было послано. Тогда $x = e + c$ и

$$xH^t = (e + c)H^t = eH^t.$$

Введем отношение эквивалентности на множестве K^n. Векторы x и y эквивалентны, если

$$x - y \in C.$$

Два вектора x и y эквивалентны, если

$$(x - y)H^t = 0.$$

Рассмотрим класс эквивалентных элементов $x + C = \{x + c \mid c \in C\}$. Синдромом этого класса называется элемент $s \in x+C$ наименьшего веса (их может быть несколько, тогда выбирем произвольный из них).

Если запомнить все синдромы и их произведение на матрицу H^t, то исправление ошибки сводится к следующей процедуре:

1. Вычислить $r = xH^t$.

2. Если $r = 0$, то ошибок не было. Иначе найти из таблицы синдром s, соответствующий r, и вернуть $x - s$.

Пример. Код Хэмминга.

Рассмотрим бинарный код. Код с проверочной матрицей, у которой в качестве столбцов берутся все двоичные представления чисел от 1 до m, называется кодом Хэмминга.

Пример.

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

– проверочная матрица для $[2^3 - 1, 2^3 - 1 - 3, 3] = [7, 4, 3]$ кода.

В случае одной ошибки синдром eH^t совпадает с номером позиции, в которой произошла ошибка.

Задачи

11.1. Найдите проверочную и порождающую матрицы линейного кода

$$C = \{(x_1, x_2, x_3, x_4) \mid x_1 + x_2 + x_3 + x_4 = 0\}.$$

Найдите кодовое расстояние кода из проверочной матрицы, число ошибок, которые код обнаруживает и исправляет. Найдите все синдромы кода и декодируйте $(1, 1, 1, 0)$.

11.2. Выпишите все кодовые слова двоичного кода, заданного порождающей матрицей. Найдите проверочную матрицу линейного кода.

$$G = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix}.$$

Найдите кодовое расстояние кода из проверочной матрицы, число ошибок, которые код обнаруживает и исправляет. Найдите все синдромы кода. Декодируйте 1111 и 0101.

11.3. Использовав бинарный код Хэмминга $[7, 4, 3]_2$, декодируйте сообщения $y_1 = (1, 1, 0, 1, 1, 0, 0)$.

11.4. Найдите проверочную матрицу линейного кода с порождающей матрицей

$$G = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \end{pmatrix}.$$

Найдите кодовое расстояние кода из проверочной матрицы, число ошибок, которые код обнаруживает и исправляет. Найдите все синдромы кода. Декодируйте 11111 и 01011.

11.5. Использовав бинарный код Хэмминга $[7, 4, 3]_2$, декодируйте сообщения $y_2 = (1, 1, 1, 1, 1, 1, 1)$, $y_3 = (1, 1, 1, 0, 0, 0, 0)$.

12 Пример проверочной работы-2

Пример второй проверочной работы в аудитории

1. Найти наибольший общий делитель чисел $a = 1716$ и $b = 627$, использовав алгоритм Евклида. Найти представление НОД в виде $d = ua + bv$.

2. Линейный код задан проверочной матрицей

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

Найти кодовое расстояние кода, число ошибок, которые код обнаруживает и исправляет, количество кодовых слов.

3. Для передачи секретной информации выбрана криптосистема RSA. Использовав открытый ключ (n, e), передать Алисе секретное сообщение m. Дешифровать его с помощью секретного ключа. Даны следующие параметры криптосистемы: $p = 17, q = 29, e = 5, m = 3$.

4. Найти произведение перестановок $\tau \cdot \sigma$. Найти порядки элементов τ, σ.

$$\tau = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 2 & 1 & 4 & 3 & 7 & 5 & 6 \end{pmatrix}; \qquad \sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 4 & 2 & 1 & 7 & 3 & 5 & 6 \end{pmatrix}.$$

Литература

[1] Sipser M. *Introduction to the Theory of Computation* / M. Sipser. – Cengage Learning, 2012. – 504 p.

[2] Kenneth H. Rosen *Discrete Mathematics and Its Applications. 5rd Edition*/Kenneth H. Rosen. – McGraw-Hill, 2003. – 1072 p.

[3] Жданов О.Н., Чалкин В.А. *Эллиптические кривые: Основы теории и криптографические приложения* / О.Н. Жданов, В.А. Чалкин. – М.: Либроком, 2020. – 200 с.

[4] *Сборник задач по теории кодирования, криптологии и сжатию данных: учебное пособие* / Ф. И. Соловьева, А. В. Лось, И. Ю. Могильных. – Новосибирск: Редакционно-издательский центр НГУ, 2013. - 99 с.