1 Регулярные выражения и конечные автоматы

1.1 Введение

В этом листке рассматриваются два объекта с очень непохожими определениями: конечные автоматы и регулярные выражения. Грубо говоря, конечный автомат — это программа на языке программирования, в котором отсутствуют массивы и есть только типы bool (возможные значения: false и true) и char (возможные значения: буквы алфавита). Программа поочерёдно читает символы из некоторой строки, длина которой заранее неизвестна, а в конце строки должна напечатать «да» или «нет». Приведём пример такой «программы», определяющей, чётное ли количество букв «a» в строке:

```
bool f = true;
while (!eof())
{
if (get_next_char() == 'a')
f = !f;
}
return f;
```

С другой стороны, регулярное выражение — это способ описывать «хитрый» поиск в тексте. Например, «в искомой строке сначала идёт от 5 до 15 букв **a**, потом несколько (возможно, ноль) раз идёт буква **b** или слово **ba**». В этих терминах можно формулировать как обычный поиск подстроки в строке, так и задачи типа «найти места, где между инициалами и фамилией не стоит пробела».

Оказывается (см. задачи **??** и **??**), что эти два понятия равносильны: для любого регулярного выражения существует конечный автомат, который определяет, найдётся ли данное выражение во входной строке, и наоборот. Это позволяет быстро выполнять сложный поиск в текстах (например, алгоритм Кнута–Морриса–Пратта поиска подстроки в строке основан именно на этой идее).

1.2 Конечные автоматы

Конечно, конечный автомат можно задать просто программой, написанной на каком-то языке программирования, но для некоторых доказательств полезна следующая конструкция. Назовём *состоянием автома-*

та состояние памяти компьютера, то есть набор значений всех используемых в программе переменных. Например, у приведённой во введении программы всего два состояния: `f=true` и `f=false`. Разобьём работу программы на участки между чтениями очередных символов из входной строки. Ясно, что состояние автомата к следующему считыванию полностью определяется состоянием к предыдущему считыванию и предыдущим символом.

Отметим теперь на плоскости точки, соответствующие различным состояниям автомата. Для каждого состояния A и буквы x проведём стрелку в то состояние, в котором окажется автомат к считыванию следующей буквы, если он в состоянии A прочёл x. Несложно видеть, что полученная картинка полностью определяет автомат. Такая картинка называется *диаграммой* автомата.

Задача 1. Нарисуйте диаграммы автоматов (или напишите их на каком-то языке программирования), распознающих следующие свойства строк:

(a) строка содержит хотя бы три буквы «a»;

(b) строка содержит или хотя бы две буквы «a» или хотя бы три буквы «b»;

(c) до первой буквы «a» в строке встречается ровно две буквы «b»;

(d) строка содержит подстроку `ababc`.

Задача 2. (a) По диаграмме автомата постройте диаграмму автомата, печатающего «да» тогда и только тогда, когда исходный печатает «нет».

(b) Решите аналогичные задачи для операций «И» и «ИЛИ».

Задача 3. Докажите, что для любого конечного автомата существует натуральное число p, такое что любое «хорошее»[1] слово длины больше p, представляется в виде $w_1w_2w_3$, так что $0 < |w_2| \le p$ и все слова вида $w_1w_2^nw_3$, $n \ge 1$ — «хорошие».

Задача 4. Какие из следующих свойств распознаются конечными автоматами:

(a) число букв `a` чётно, а число букв `b` нечётно;

(b) количество букв `a` равно количеству букв `b`;

(c) слово не содержит подслова `bad word`;

(d) количество букв `a` в два раза больше количества букв `b`;

(e) слово отличается от данного не более, чем в одной букве.

[1] слово, для которого рассматриваемый автомат печатает «да»

1.3 Регулярные выражения

Как было сказано, регулярные выражения позволяют делать «хитрые» запросы по поиску в тексте.

Формально регулярное выражение и описываемое им свойство определяется индуктивно:

1. пустая строка — регулярное выражение, соответствующее только пустой строке;
2. $\varnothing$ — регулярное выражение, под которое не подходит ни одна строка;
3. каждая буква алфавита — регулярное выражение, под которое подходит только строка из одной этой буквы;
4. если R_1 и R_2 — регулярные выражения, то:
   - $(R_1|R_2)$ — регулярное выражение, под которое подходят строки, подходящие или под R_1 или под R_2;
   - R_1R_2 — регулярное выражение, под которые подходят строки вида w_1w_2, где w_1 подходит под R_1, а w_2 — под R_2;
   - R_1^* — регулярное выражение, под которое подходят пустая строка и строки вида $w_1w_2\dots w_n$, где каждая из строк w_k подходит под R_1.

Задача 5. Опишите, какие слова удовлетворяют следующим регулярным выражениям:
(a) a^*a; (b) $a^*ba^*ba^*$; (c) $(a|ba)^*$; (d) $(a^*ba^*)^*$; (e) $a^*(ba^*ba^*)^*$.

Задача 6. Задайте регулярными выражениями следующие множества слов:
(a) слова, не содержащие подслова aba;
(b) текст в скобках, не содержащий внутри себя других скобок;
(c) слова из букв a, b, c, в которых чётность количества букв a совпадает с чётностью количества букв b.

Задача 7. Докажите, что для каждого регулярного выражения R существует регулярное выражение R', такое что слово w удовлетворяет выражению R' тогда и только тогда, когда слово, получающееся из w переписыванием справа налево, удовлетворяет выражению R.

Задача 8. Докажите аналог леммы о разрастании для регулярных выражений.

Задача 9. Докажите, что для каждого регулярного выражения существует конечный автомат, распознающий слова, задаваемые этим выражением, и только их.

Задача 10. Докажите, что для каждого конечного автомата существует регулярное выражение, задающее слова, распознаваемые конечным автоматом, и только их.