Практика 7-8

Конечные автоматы
(Алфавиты, слова, языки.)

Мы уже говорили, что выполняя вычислительный процесс, компьютер работает с ***текстами***, которые представляют собой последовательность символов некоторого алфавита. И входные, и выходные данные, а также сама программа являются **текстами над алфавитом.** Этот алфавит состоит из символов клавиатуры компьютера.

*К*омпьютеры на атомарном уровне оперируют битами и регистрами. Люди изъясняются на естественных языках или пользуются математическими обозначениями. Совместить это можно с помощью искусственного языка, позволяющего употреблять строго определенное множество слов, предложений и формул, понятных машине. В языке определяются допустимые конструкции и значения, а для ЭВМ создается ПО, с помощью которого машина может воспринимать последовательности битов, представляющие команды или программы, написанные на искусственных языках и переведенные в коды машины. Языки общения с ЭВМ различны это :

- машинные языки - набор команд конкретной машины, который интерпретируется на аппаратном уровне;
- языки ассемблера – языки низкого уровня, в значительной мере отражающие набор команд конкретной машины;
- языки высокого уровня, имеющие сложную структуру и не зависящие от набора команд и ОС.

Существуют программы для обработки языков –

- интерпретаторы - программы, которые допускают на входе исходную программу, записанную на исходном языке, и производят вычисления, предписываемы этой программой. Таким образом, интерпретатор отвечает за исполнение программы. Интерпретаторы удобны при отладке.

Например, интерпретаторы языков Lisp, Scheme, Python. В Lisp интерпретатор считывает законченную конструкцию языка, то есть S-выражение, выполняет ее, печатает результаты, после чего переходит в режим ожидания ввода пользователем следующей конструкции языка.

- Компилятор читает сразу всю программу и переводит ее в объектный код, который непосредственно выполняется компьютером. Объектный код также называют двоичным или машинным кодом. Когда программа скомпилирована, в ее коде уже нет отдельных строк исходного кода.

В общем случае интерпретируемая программа выполняется медленнее, чем скомпилированная. Потеря времени на компиляцию происходит лишь единожды, а в случае интерпретации — каждый раз при очередной компиляции фрагмента программы в процессе ее выполнения.

Java, например, рассчитан в основном на интерпретацию программы, а язык C — на компиляцию.

В основе всех процессов обработки языков лежит теория автоматов и формальных языков. При этом *под автоматом подразумевается не реально существующее устройство, а математическая модель, свойства и поведение которой можно имитировать с помощью программы на реальной вычислительной машине.* Конечные автоматы моделируют вычислительные устройства с фиксированным и конечным объемом памяти, которые читают последовательности входных символов, принадлежащих некоторому конечному множеству.

Конечный автомат является простейшей вычислительной моделью. Он способен решать задачи без использования каких-либо вспомогательных переменных, то есть не имеет памяти.

При этом входное слово доступно только для чтения, только один раз и только слева направо, а результат работы получается сразу после прочтения последнего входного символа.

Так как при моделировании конечного автомата на компьютере обработка одного входного символа требует небольшого количества операций, программа работает очень быстро. Конечный автомат помогает нам понять, как происходит моделирование процесса вычислений.

Описывая работу конечного автомата мы вводим необходимые понятия конфигурации, шага вычисления, процесса вычисления вообще, детерминизма и недетерминизма.

Основные вопросы, на которые мы должны ответить, определяя вычислительную модель:

- Набор допустимых элементарных операций, то есть набор команд, которые можно использовать для построения программы.
- Операции доступа к памяти и операции работы с ней .
- Как в компьютер поступают входные данные.
- Как определить выход (результат).
-

Но мы сказали, что памяти у нас нет. На самом деле кроме той, в которой записана сама программа.

Есть также указатель на текущую строку выполнения (команду программы).

Идея определения конечных автоматов состоит в том, что содержание указателя – то есть номер фактически выполняемой строки программы – это и есть изменяемая информация, и программа работает с такой псевдопеременной.

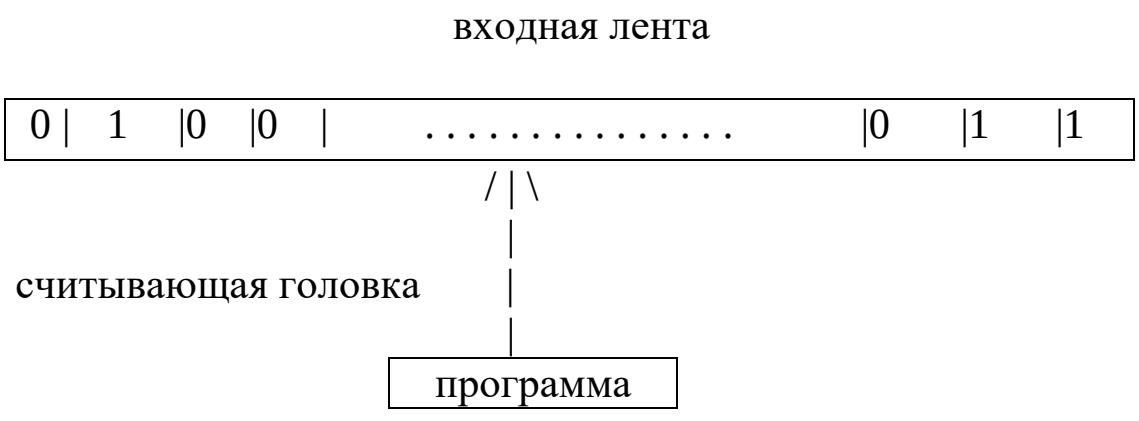

Рис. Конечный автомат

Основные компоненты конечного автомата (вычислительной модели)

- Программа, записанная в памяти (другой памяти – нет).
- Входная лента, содержащая входное слово.
- Считывающая головка, способная перемещаться по входной ленте слева направо.

Конечный автомат не имеет никакой памяти – кроме той, в которую записана сама программа, а также указателя на выполняющуюся в текущее время строку (команду) программы. Это значит, что программе не разрешено использовать переменные. А как же ей вычислять что – либо без использования переменных? Но в этом –то и состоит идея определения конечных автоматов. Содержание указателя – то есть номер фактически выполняемой строки программы – это и есть изменяемая информация, и программа работает с этой псевдопеременной. Программа читает очередной входной символ, сравнивает его с алфавитом и если входной символ совпадает с некоторой буквой алфавита, то программа продолжает работать с этой строкой. Каждая строка программы содержит в точности одну вышеописанную команду.
Такие программы можно рассматривать как алгоритмы для решения специального варианта проблемы принадлежности , так как они отвечают на вопрос, принадлежит ли некоторое слово заданному языку. Все строки пронумерованы 0,1,2,3,. . . , и поэтому на каждом шаге вычисления ответ полностью определяется номером строки.

select input=a_1 goto i_1
input=a_2 goto i_2
.
.
input=a_k goto i_r

Если заданный алфавит {a_1, a_2, ...,a_k }, то конечный автомат может использовать только вышеописанные команды.

Входную ленту можно рассматривать как линейную память для ввода. Каждая ее ячейка является единицей памяти и содержит в точности один символ алфавита. По факту лента будет содержать ровно столько ячеек, сколько символов содержится во входном слове.

Конечным автоматом (КА) называется формальная система:

$$M = (Q, \Sigma, \delta, q_0, F), \text{ где}$$

- Σ - конечное множество входных символов (конечный входной алфавит),
- Q - конечное непустое множество состояний,
- δ - функцией перехода, которая каждой паре , состоящей из входного символа и текущего состояния, приписывает новое состояние (отображение типа $Q^* \Sigma \rightarrow Q$),
- $q_0 \in Q$- выделенное начальное состояние
- $F \subseteq Q$ – подмножество состояний, выделенных в качестве допускающих или заключительных.

Запись $\delta(q,a)=p$, где $p,q \in Q$ и $a \in \Sigma$, означает, что КА в состоянии q, сканируя входной символ a , продвигает свою входную головку на одну ячейку вправо и переходит в состояние p.

Определенная таким образом модель КА называется детерминированной .

Определение Цепочка $x \in \Sigma$ принимается КА M, если $\delta(q_0, x)=p$ для некоторого $p \in F$.
Множество всех цепочек $x \in \Sigma$, принимаемых конечным автоматом M, называется языком, распознаваемым КА M, и обозначается как L(M).

$$L(M)= \{ x \in \Sigma \mid \delta(q_0, x)=p \text{ при некотором } p \in F \}$$

Рассмотрим некоторый конечный распознаватель – модель с конечным числом состояний, которое отличает правильно образованные цепочки – допустимые от недопустимых. С точки зрения математики мы можем определить это в терминах множеств, последовательностей (цепочек) и функций. Но можем представить и в виде вычислительной машины.

Проверка нечетности числа единиц в произвольной цепочке, состоящей из нулей и единиц. Конечный автомат будет допускать все цепочки, содержащие нечетное число единиц, и отвергать цепочки с четным их числом. (контроллер нечетности)

На вход конечного автомата подается цепочка символов из конечного множества, называемого входным алфавитом автомата Входной алфавит – это совокупность символов, для работы с которыми он предназначен. Как допускаемые, так и отвергаемые автоматом цепочки состоят только из символов входного алфавита. Символы, не принадлежащие входному алфавиту, нельзя подавать на вход автомата. Входной алфавит контроллера нечетности состоит из двух символов - 0 и 1.

В каждый момент времени конечный автомат имеет дело лишь с одним входным символом, а информация о предыдущих символах входной цепочки сохраняется с помощью конечного множества состояний. Это значит, что автомат помнит о прочитанных ранее символах только то, что при их обработке он перешел в некоторое состояние, которое и есть его память о прошлом.

Наш контроллер нечетности устроен так, что он запоминает, четное или нечетное число единиц встретилось ему при чтении отрезка входной цепочки. Поэтому множество состояний нашего автомата содержит 2 состояния – ЧЕТ и НЕЧЕТ

Одно из этих состояний должно быть выбрано как начальное. Пусть это будет ЧЕТ. При чтении очередного символа его состояние меняется. Новое состояние зависит только от входного символа и текущего состояния. Такое изменение состояния будем называть ПЕРЕХОДОМ. При этом новое состояние может совпасть со старым.

Работу автомата можно описать Σ, L с помощью функции перехода δ.

По текущему состоянию $s_{тек}$ и текущему входному символу x δ дает новое состояние автомата:

$$\delta(s_{тек}, x) = s_{нов}$$

Это можно изобразить более наглядно так:

$$S_{тек} \xrightarrow{x} S_{нов}$$

Определим функцию переходов контроллера:

δ(ЧЕТ, 0) = ЧЕТ //четность меняется тогда и только тогда, когда на входе читается единица
δ(ЧЕТ, 1) = НЕЧЕТ // на входе читается 1
δ(НЕЧЕТ, 0) = НЕЧЕТ
δ(НЕЧЕТ, 1) = ЧЕТ

Некоторые состояния автомата выбираются как допускающие или ЗАКЛЮЧИТЕЛЬНЫЕ .

Если автомат, начав работу в начальном состоянии, при прочтении всей цепочки переходит в одно из допускающих состояний, то говорят, что входная цепочка допускается автоматом. Если последнее состояние автомата не является допускающим, говорят, что автомат отвергает цепочку .

Контроллер нечетности имеет единственное допускающее состояние –НЕЧЕТ.

Для контроллера нечетности переход автомата из текущего состояния в новое можно записать так:

$$НЕЧЕТ \xrightarrow{1} ЧЕТ$$

Допустим, есть цепочка 1 1 0 1

1 1 0 1
ЧЕТ –> НЕЧЕТ –> ЧЕТ –> ЧЕТ –> НЕЧЕТ

Автомат в состоянии ЧЕТ применяется к цепочке 1 1 0 1 .

Так как ЧЕТ –начальное, а НЕЧЕТ - допускающее состояние, цепочка 1 1 0 1 допускается нашим автоматом.

? Что произойдет, если входная цепочка будет иметь вид 1 0 1
Изобразим это символически

1 0 1
ЧЕТ---НЕЧЕТ----НЕЧЕТ----ЧЕТ (автомат отвергнет эту цепочку)

Таблица переходов – один из способов представления конечных автоматов

	0	1	
чет	чет	нечет	0
нечет	нечет	чет	1

Таблица переходов для контроллера

1. Столбцы помечены входными символами.
2. Строки помечены символами состояний
3. Элементами таблицы являются символы новых состояний, соответствующих входным символам столбцов и состояниям строк
4. Первая строка помечена символом начального состояния

Строки, соответствующие допускающим состояниям, помечены справ единицами, а строки, соответствующие отвергающим состояниям, помечены справа нулями

Таблица задает конечный автомат , у которого :

Входное множество ={0,1}, Множество состояний ={ ЧЕТ, НЕЧЕТ}, Переходы δ(ЧЕТ,0)=ЧЕТ и т.д. Начальное состояние = ЧЕТ, Допускающее состояние ={НЕЧЕТ}

Другим неформальным описанием программы M является ее графическое представление G(M)/

Для определения КА можно его программе M поставить в соответствие помеченный ориентированный граф G(M) .Этот граф имеет столько вершин, сколько строк содержит программа M.

Каждая вершина соответствует одной строке автомата, а номер этой строки является пометкой рассматриваемой вершины. И если программа M переходит из строки i в строку j, читая входной символ x, то G(M) содержит ориентированную дугу (i, j), помеченную таким x.

Пример. Рассмотрим диаграмму состояний конечного автомата. Пусть задан конечный автомат $M = (Q, \Sigma, \delta, q_0, F)$, где $Q = \{q_0, q_1, q_2, q_3\}$, $\Sigma = \{0, 1\}$, $F = \{q_0\}$, $\delta(q_0, 0) = q_2$, $\delta(q_0, 1) = q_1$, $\delta(q_1, 0) = q_3$, $\delta(q_1, 1) = q_0$,
$\delta(q_2, 0) = q_0$, $\delta(q_2, 1) = q_3$, $\delta(q_3, 0) = q_1$, $\delta(q_3, 1) = q_2$.

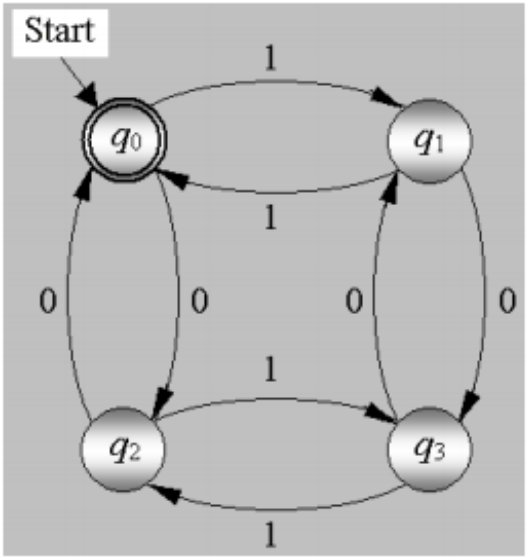

Диаграмма состояний КА состоит из узлов, представляющих состояния, и из ориентированных дуг, определяющих возможные переходы, которые зависят от входных символов.
Например, если $\delta(q, x)=p$, то из узла, представляющего состояние q, в узел, представляющий состояние p, проводится дуга, помеченная входным символом x.

На рисунке показана диаграмма состояний КА M. Двойным кружком выделено единственное в данном случае допустимое состояние, которое является одновременно и начальным состоянием.

Пусть на входе действует цепочка 1 1 0 1 0 1.

Просканировав входную цепочку $\delta(q_0, 110101)=q_0$, мы видим, что цепочка находится в языке T(M). Убедитесь в этом.

На самом деле , T(M) есть множество всех цепочек из {0,1} , содержащих четное число нулей и четное число единиц.

Графическое представление программы дает однозначный и ясный способ описания автомата.

Итак, конечные автоматы соответствуют подклассу простых алгоритмов для решения проблемы распознавания цепочек, в общем случае – распознавания языков.

Конечный автомат не использует никаких переменных, то есть никакой памяти. КА использует движение среди конечного числа состояний(строк программы) и принимает входное слово если после прочтения всего слова работа автомата завершается в одном из допустимых состояний.
Множество L всех слов ,допускаемых программой- автоматом, является языком, допускаемым этим автоматом. Это значит, что конечный автомат решает проблему принадлежности (принятия) – проблему (Σ, L) для распознаваемого языка L.

Теория автоматов лежит в основе построения компиляторов.

В дальнейшем мы будем рассматривать машину Тьюринга как обобщение конечного автомата. Машина Тьюринга сможет использовать входную ленту не только для того, чтобы читать, но и как память, записывая в нее необходимые символы.

Самостоятельная работа

Задача 49 Таблица задает КА, у которого

	x	y	z	
1	1	3	4	1
2	2	1	3	0
3	2	4	4	1
4	3	3	3	0

Входное множество ={x, y, z},
Множество состояний ={ 1, 2, 3, 4},
Переходы $\delta(1,x)=1$, $\delta(1, y) = 3$ и т.д.
Начальное состояние = 1
Допускающее состояние ={1, 3}

Действуют две входные цепочки :

1) xyzz
2) zyx

Определите, какая из цепочек допускается автоматом, а какая не допускается. Или, может быть, обе цепочки допускаются, или наоборот, не допускаются.

1. Нарисуйте последовательность переходов.
2. Дайте графическое представление этого конечного автомата.

Задача 40 Таблица задает КА, у которого

	x	y	z	
1	1	3	4	1
2	2	1	3	0
3	2	4	4	1
4	3	3	3	0

Входное множество ={x, y, z},
Множество состояний ={ 1, 2, 3, 4},
Переходы $\delta(1,x)=1$, $\delta(1, y) = 3$ и т.д.
Начальное состояние = 1
Допускающее состояние ={2, 3}

Действуют две входные цепочки :

1) zyx
2) xyzz

Определите, какая из цепочек допускается автоматом, а какая не допускается. Или, может быть, обе цепочки допускаются, или наоборот, не допускаются.

1.Нарисуйте последовательность переходов

2 Дайте графическое представление этого конечного автомата.

Задача 43 Таблица задает КА, у которого

	x	y	z	
1	1	3	4	1
2	2	1	3	0
3	2	4	4	1
4	3	3	3	0

Входное множество ={x, y, z},
Множество состояний ={ 1, 2, 3, 4},
Переходы $\delta(1,x)=1$, $\delta(1, y) = 3$ и т.д.
Начальное состояние = 1
Допускающее состояние ={1, 3}

Действуют две входные цепочки :

1) xzy
2) xxz

Определите, какая из цепочек допускается автоматом, а какая не допускается. Или, может быть, обе цепочки допускаются, или наоборот, не допускаются.

1.Нарисуйте последовательность переходов

2.Дайте графическое представление этого конечного автомата.

Задача 42 Таблица задает КА, у которого

	x	y	z	
1	1	3	4	1
2	2	1	3	0
3	2	4	4	1
4	3	3	3	0

Входное множество ={x, y, z},
Множество состояний ={ 1, 2, 3, 4},
Переходы $\delta(1,x)=1$, $\delta(1, y) = 3$ и т.д.
Начальное состояние = 1
Допускающее состояние ={2, 4}

Действуют две входные цепочки :

1) yxy
2) yzy

Определите, какая из цепочек допускается автоматом, а какая не допускается. Или, может быть, обе цепочки допускаются, или наоборот, не допускаются.

1.Нарисуйте последовательность переходов
2.Дайте графическое представление этого конечного автомата.

Задача 41 Таблица задает КА, у которого

	x	y	z	
1	1	3	4	1
2	2	1	3	0
3	2	4	4	1
4	3	3	3	0

Входное множество ={x, y, z},
Множество состояний ={ 1, 2, 3, 4},
Переходы $\delta(1,x)=1$, $\delta(1, y) = 3$ и т.д.
Начальное состояние = 3
Допускающее состояние ={1, 4}

Действуют две входные цепочки :

1) yxy
2) yzy

Определите, какая из цепочек допускается автоматом, а какая не допускается. Или, может быть, обе цепочки допускаются, или наоборот, не допускаются.

1.Нарисуйте последовательность переходов

2.Дайте графическое представление этого конечного автомата.