Конечный автомат

W ru.wikipedia.org/wiki/Конечный_автомат

Конечный автомат (КА) — (в теории алгоритмов) — математическая абстракция, модель дискретного устройства, имеющего один вход, один выход и в каждый момент времени находящегося в одном состоянии из множества возможных. Является частным случаем абстрактного дискретного автомата, число возможных внутренних состояний которого *конечно*.

Абстрактный автомат

При работе на вход КА поступают последовательно входные воздействия, а на выходе КА формирует выходные сигналы. Обычно под входными воздействиями принимают подачу на вход автомата символов одного алфавита, а на выход КА в процессе работы выдаёт символы в общем случае другого, возможно даже не пересекающегося со входным, алфавита.

Помимо конечных автоматов существуют и бесконечные дискретные автоматы — автоматы с бесконечным числом внутренних состояний, например, машина Тьюринга.

Переход из одного внутреннего состояния КА в другое может происходить не только от внешнего воздействия, но и самопроизвольно.

Различают детерминированные КА — автоматы, в которых следующее состояние однозначно определяется текущим состоянием и выход зависит только от текущего состояния и текущего входа, и недетерминированные КА, следующее состояние у которых в общем случае неопределённо и, соответственно, не определён выходной сигнал. Если переход в последующие состояния происходит с некоторыми вероятностями, то такой КА называют *вероятностным* КА.

Примерами физической реализации КА могут служить любые цифровые системы, например, компьютеры или некоторые логические узлы компьютеров с памятью — триггеры и другие устройства. Комбинационная последовательная логика не может являться КА, так как не имеет внутренних состояний (не имеет памяти).

С абстрактной точки зрения КА изучает раздел дискретной математики — *теория конечных автоматов*.

Теория конечных автоматов практически широко используется, например, в синтаксических и лексических анализаторах, тестировании программного обеспечения на основе моделей.

Формально КА определяется как пятёрка:

$A = (S , X , Y , \delta , \lambda) , {\displaystyle A=(S,X,Y,\delta ,\lambda),}$

где S {\displaystyle S} — конечное множество состояний автомата;

X , Y {\displaystyle X,Y} — конечные входной и выходной алфавиты соответственно, из которых формируются строки, считываемые и выдаваемые автоматом;

$\delta : S \times X \rightarrow S$ {\displaystyle \delta :S\times X\rightarrow S} — функция переходов;

$\lambda : S \times X \rightarrow Y$ {\displaystyle \lambda :S\times X\rightarrow Y} — функция выходов.

Абстрактный автомат с некоторым выделенным состоянием $s o) , {\displaystyle s_{o}),}$, это состояние называют *начальным состоянием*, называется *инициальным автоматом*. Так как каждый КА имеет конечное число состояний, и любое из его состояний может быть назначено начальным состоянием, одному и тому же автомату соответствует N {\displaystyle N} инициальных автоматов, N {\displaystyle N} — число внутренних состояний КА. Таким образом, абстрактный КА определяет семейство инициальных автоматов. Если не указано начальное состояние, то поведение автомата всегда недетерминировано, выходное слово автомата зависит от начального состояния, поэтому полное детерминированное описание автомата будет[1]:

$A = (S , X , Y , \delta , \lambda , s o) . {\displaystyle A=(S,X,Y,\delta ,\lambda ,s_{o}).}$

Различают два класса КА: автоматы Мура — КА, у которых выходной сигнал зависит только от внутреннего состояния, по рисунку у автомата Мура нет связи от входа $x (t) {\displaystyle x(t)}$ к функции выхода λ {\displaystyle \lambda } и автоматы Мили — выходной сигнал зависит как от внутреннего состояния, так и от состояния входа.

Общее описание

Существуют различные способы задания алгоритма функционирования конечного автомата. Например, конечный автомат может быть задан в виде упорядоченной пятерки элементов некоторых множеств:

$M = (V , Q , q o , F , \delta) , {\displaystyle M=(V,Q,q_{o},F,\delta),}$

где V {\displaystyle V} — *входной алфавит* (конечное множество *входных символов*), из которого формируются *входные слова*, воспринимаемые конечным автоматом;

Q {\displaystyle Q} — *множество внутренних состояний*;

$q o$ {\displaystyle q_{o}} — *начальное состояние* $(q o \in Q)$ {\displaystyle (q_{o}\in Q)} ;

F {\displaystyle F} — множество *заключительных, или конечных состояний* (F ⊂ Q) {\displaystyle (F\subset Q)} - ;

δ {\displaystyle \delta } — *функция переходов*, определённая как отображение δ : Q × (V ∪ { ε }) → Q {\displaystyle \delta \colon Q\times (V\cup \{\varepsilon \})\rightarrow Q} , такое, что δ (q , a) = { r : q → a r } {\displaystyle \delta (q,a)=\{r\colon q\,\,{\underset {a}{\to }}\,\,r\}} , то есть значение функции переходов на упорядоченной паре (состояние, входной символ или пустая цепочка символов) есть множество всех состояний, в которые из данного состояния возможен переход по данному входному символу или пустой цепочке символов, обычно обозначаемой буквой ε . {\displaystyle \varepsilon .}

При анализе КА принято полагать, что конечный автомат начинает работу в некотором начальном состоянии q 0 {\displaystyle q_{0}} , последовательно получает по одному символу из входного слова (цепочки входных символов). Считанный символ может перевести автомат в новое состояние или не перевести в новое состояние в соответствии с функцией переходов.

Получая входную цепочку символов x {\displaystyle x} и делая переходы из состояния в состояние, автомат после получения последнего символа[*прояснить*]. входного слова окажется в некотором состоянии q ′ {\displaystyle q'} .

Если это состояние является заключительным, то говорят, что автомат допустил слово x {\displaystyle x} [*прояснить*]

Другие способы задания функционирования КА

	Следующее состояние		
Исходное состояние	**Входной символ a**	**Входной символ b**	**Любой другой символ**
p0	p1	p0	p0
p1	p1	p2	p1
p2	p3	p4	p2
p3	p3	p5	p3
p4	p4	p4	p4
p5	p3	p5	p5

1. **Диаграмма состояний** (или иногда **граф переходов**) — графическое представление множества состояний и функции переходов. Представляет собой размеченный ориентированный граф, вершины которого — состояния КА, дуги — переходы из одного состояния в другое, а *метки дуг* — символы, по которым осуществляется переход из одного состояния в другое. Если переход из состояния q_1 в q_2 может быть осуществлен по одному из *нескольких* символов, то все они должны быть надписаны над дугой диаграммы.
2. **Таблица переходов** — табличное представление функции δ. Обычно в такой таблице каждой строке соответствует одно состояние, а столбцу — один допустимый входной символ. В ячейке на пересечении строки и столбца записывается состояние, в которое должен перейти автомат, если в данном состоянии он считал данный входной символ. Пример таблицы переходов для автомата, заданного в виде графа по рисунку 1 приведена справа.

Детерминированность

Конечные автоматы

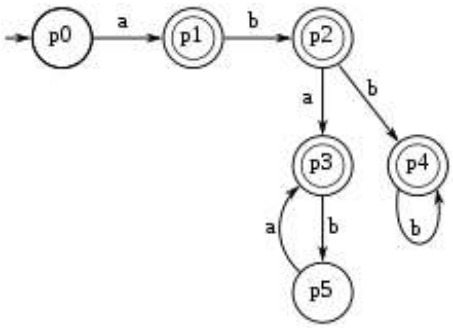


Рисунок 1. Пример графа переходов детерминированного КА.

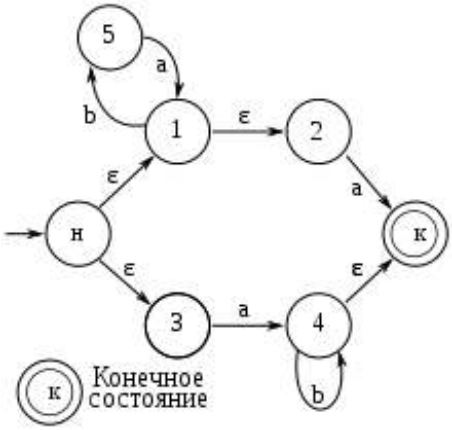
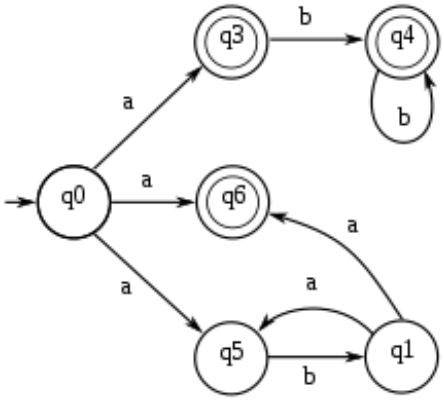


Рисунок 2. Пример графа переходов недетерминированного КА с самопроизвольными переходами

подразделяются на детерминированные и недетерминированные.



- Детерминированным конечным автоматом (ДКА) называется такой автомат, в котором нет дуг с меткой ε (предложение, не содержащее ни одного символа), и из любого состояния по любому символу возможен переход не более, чем в одно состояние.

- Недетерминированный конечный автомат (НКА) является обобщением детерминированного. Недетерминированность автоматов может достигаться двумя способами: либо могут существовать переходы из состояние в состояние вызываемые пустой цепочкой символов, то есть самопроизвольные переходы без внешних воздействий, либо из одного состояния КА может переходит в разные состояния под воздействием одного и того же символа.

Рисунок 3. Недетерминированный КА с переходами из одного состояния в разные состояния при одинаковом входном воздействии

Если рассмотреть случай, когда автомат формально задан следующим образом: M = (V , Q , S , F , δ) {\displaystyle M=(V,Q,S,F,\delta)} , где S {\displaystyle S} — множество *начальных состояний* автомата, такое, что S ⊆ Q {\displaystyle S\subseteq Q} , то появляется третий признак недетерминированности — наличие нескольких начальных (стартовых) состояний у автомата M {\displaystyle M} .

Теорема о детерминизации утверждает, что для любого конечного автомата может быть построен эквивалентный ему детерминированный конечный автомат (два конечных автомата называют эквивалентными, если их языки совпадают[*прояснить*]). Однако поскольку количество состояний в эквивалентном ДКА в худшем случае растёт экспоненциально с ростом количества состояний исходного НКА, на практике подобная детерминизация не всегда возможна. Кроме того, конечные автоматы с выходом в общем случае не поддаются детерминизации.

В силу последних двух замечаний, несмотря на □ большую сложность недетерминированных конечных автоматов, для задач, связанных с обработкой текста, преимущественно применяются именно НКА.

Автоматы и регулярные языки

Для конечного автомата можно определить язык (множество слов) в алфавите V {\displaystyle V} , который он **допускает** — так называются слова, чтение которых переводит автомат из начального состояния в одно из заключительных состояний.

Теорема Клини утверждает, что язык является регулярным тогда и только тогда, когда он допускается некоторым конечным автоматом, используемым в этом языке.

Минимизация автоматов

Основная статья: **Минимизация ДКА**

Для любого регулярного языка существует единственный с точностью до изоморфизма автомат, принимающий этот язык и обладающий при этом наименьшим возможным числом состояний. Минимальный автомат для языка, заданного детерминированным конечным автоматом может быть осуществлена за полиномиальное время, что позволяет оптимизировать расход памяти, требуемый для работы с автоматом, а также решать такие задачи, как проверка эквивалентности двух автоматов за полиномиальное время.

Специализированные языки программирования

> Язык последовательных функциональных схем SFC (Sequential Function Chart) — графический язык программирования, широко используется для программирования промышленных логических контроллеров (ПЛК).

В графическом языке SFC программа описывается в виде схематической последовательности шагов, объединённых переходами.

Конечные автоматы позволяют построить модели систем параллельной обработки, однако, чтобы изменить число параллельных процессов в такой модели требуется внести существенные изменения в саму модель. Кроме того, попытка разработки сложной модели реализованной конечным автоматом приводит к быстрому росту числа состояний автомата, что в итоге сделает разработку такой модели крайне трудоёмкой. Как было отмечено выше, последнюю проблему можно решить, если использовать недетерминированный автомат.

Ответ дается в различных терминах в зависимости от того, является ли автомат (соответственно П-машина) автономным или нет[2]. Автономный конечный автомат, начиная с некоторого такта, может лишь генерировать периодическую последовательность состояний **x** (соответственно П-машина — последовательность выходных символов **y**). Если эта последовательность состоит лишь из одного символа, то это означает, что за конечное число тактов автомат достигает равновесного состояния. Если же эта последовательность содержит несколько символов, это означает, что автомат последовательно проходит состояния, соответствующие этим символам, а затем работа автомата неограниченно долго периодически повторяется. Более того, какова бы ни была периодическая последовательность состояний конечной длины, всегда может быть построен автономный конечный автомат, который, начиная уже со второго такта, генерирует эту последовательность. Ничего иного, кроме периодического повторения одного и того же состояния или конечной последовательности состояний, автономный автомат «делать» не может. Однако в связи с тем, что последовательное выполнение заданного цикла операций типично для многих областей современной техники, динамические системы, которые в приемлемой идеализации можно рассматривать как автономный автомат, имеют широкое применение.

Классическим примером могут служить автоматы-куклы, выполнявшие сложные последовательности действий, например: пишущие на бумаге определённый текст, играющие на рояле заранее установленные пьесы т. д.

Современным примером служат многие станки-автоматы, автоматические линии и системы автоматического управления циклическими производствами. Если автомат не автономен, то есть состояние входа изменяется от такта к такту, то ответ на вопрос, что может «делать» и что не может «делать» конечный автомат, можно дать в разных терминах. Например, ответ можно сформулировать на языке представления событий. Действительно, неавтономный конечный автомат или последовательностная машина лишь преобразуют входные последовательности символов в последовательности состояний или выходных символов, и сказать, что может и что не может «делать» конечный автомат, значит выяснить, какие преобразования последовательностей возможны в конечном автомате, а какие невозможны. Но так как количество состояний (соответственно выходных символов) конечно, этот вопрос эквивалентен такому вопросу: при каких входных последовательностях возникает каждое из возможных состояний (или каждый из выходных символов). Этот последний вопрос в терминах, принятых в теории конечных автоматов, формулируется так: какие события могут и какие не могут быть представлены в конечном автомате каждым из возможных состояний (или каждым из выходных символов).

Ответ дается теоремами Клини. Этот ответ точный, так как теоремы Клини устанавливают необходимые и достаточные условия представимости последовательности событий в автомате, а именно: выделяются особые множества последовательностей входных символов — регулярные множества. Факт появления входной последовательности из такого множества называется соответствующим регулярным событием. Теоремы Клини устанавливают, что в конечном автомате могут быть представлены регулярные события и только они. Таким образом, на языке представления событий ответ на вопрос, что может «делать» конечный автомат, дается однозначно: конечный автомат может представлять только регулярные события. Ряд важных множеств входных последовательностей, с которыми часто приходится иметь дело на практике, заведомо регулярны. Так, например, заведомо регулярно множество, состоящее из любого конечного числа входных последовательностей конечной длины; множество любых периодических входных последовательностей; множество бесконечных последовательностей, которое содержит заданные конечные последовательности на протяжении нескольких последних тактов, и т. д.

В общем случае, если каким-либо произвольным способом задано бесконечное множество входных последовательностей, то остается открытым вопрос о том, регулярно ли это множество. Дело в том, что понятие регулярного множества вводится индуктивно, то есть устанавливается алгоритм построения любых

регулярных множеств. Однако, не существует достаточно эффективного способа решения обратной задачи, то есть установления того, является ли каждое заданное множество регулярным.

Хотя теоремы Клини и отвечают на вопрос о том, что может делать конечный автомат, но отвечают они на этот вопрос неэффективно. Сделаны первые попытки построения иных языков, на которых ответ может быть дан эффективно. Эта проблема языка, играющая кардинальную роль в получении эффективного ответа на вопрос, что может и что не может «делать» конечный автомат, имеет решающее значение и для первых этапов синтеза автомата, то есть для ответа на второй из сформулированных выше вопросов. Если расширить класс динамических систем, которые мы определили терминами «конечный автомат» и «последовательностная машина», включением бесконечной памяти (моделью бесконечной памяти может быть, например, бесконечная лента для хранения символов или бесконечное число состояний), то для динамических систем этого более широкого класса (абстрактные системы этого класса называют машинами Тьюринга) ответ на вопрос «что они могут делать?» значительно проще — они могут реализовать **любой наперед заданный алгоритм**. При этом само понятие алгоритма трактуется в современной математике как реализация вычисления значений какой-либо рекурсивной функции. Столь однозначный и четкий ответ на вопрос «что может делать машина Тьюринга?» дает возможность положить понятие о машине Тьюринга в основу определения понятия алгоритма: алгоритмом называется любой процесс, который может быть осуществлен на конечном автомате, дополненном бесконечной памятью, то есть алгоритмически полных машинах, на машине Тьюринга, на машине Поста и др.

Примечания

1. ↑ *Кузнецов О. П., Адельсон-Вельский Г. М.* Автоматы // Дискретная математика для инженера. — М. (Москва): Энергия, 1980. — 344 с.
2. ↑ *Айзерман М. А., Гусев Л. А., Розоноэр Л. И., Смирнова И. М., Таль А. А.* Логика. Автоматы. Алгоритмы. Гос. изд. физ.-мат. литературы 1963, 556 стр.

Литература

- *Белоусов А. И., Ткачев С. Б.* Дискретная математика. — М. (Москва): МГТУ, 2006. — С. 460—587. — ISBN 5-7038-2886-4.
- *Джон Хопкрофт, Раджив Мотвани, Джеффри Ульман.* Дискретная математика. — 2-е изд. — Вильямс, 2002. — 528 с. — (Алгоритмы и методы. Искусство программирования).
- *Серебряков В. А., Галочкин М. П., Гончар Д. Р., Фуругян М. Г.* Теория и реализация языков программирования — М.: МЗ-Пресс, 2006 г., 2-е изд. — ISBN 5-94073-094-9

- Теория автоматов / *Э. А. Якубайтис, В. О. Васюкевич, А. Ю. Гобземис, Н. Е. Зазнова, А. А. Курмит, А. А. Лоренц, А. Ф. Петренко, В. П. Чапенко* // Теория вероятностей. Математическая статистика. Теоретическая кибернетика. — М.: ВИНИТИ, 1976. — Т. 13. — С. 109—188. — URL http://www.mathnet.ru/php/getFT.phtml?jrnid=intv&paperid=28&what=fullt&option_lang=rus
- Применение конечных автоматов для решения задач автоматизации
- *Глушков В. М.* Синтез цифровых автоматов. — М.: ГИФМЛ, 1962. — 476 с.

Ссылки

- *Дехтярь М. И.* Введение в схемы, автоматы и алгоритмы
- Open source генератор конечных автоматов на языках C++ и Java по XML файлам описания
- Недетерминированные конечные автоматы
- *Подзоров С. Ю.* Курс лекции по теории алгоритмов