ВикипедиЯ

Конечный автомат

Материал из Википедии — свободной энциклопедии

Коне́чный автома́т — абстрактный автомат, число возможных внутренних состояний которого конечно.

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

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

где

- V — *входной алфавит* (конечное множество *входных символов*), из которого формируются *входные слова*, воспринимаемые конечным автоматом;
- Q — *множество внутренних состояний*;
- q_0 — *начальное состояние* $(q_0 \in Q)$;
- F — множество *заключительных, или конечных состояний* $(F \subset Q)$;
- δ — *функция переходов*, определенная как отображение $\delta: Q \times (V \cup \{\varepsilon\}) \to Q$, такое, что $\delta(q, a) = \{r: q \underset{a}{\to} r\}$, то есть значение функции переходов на упорядоченной паре (состояние, входной символ или пустая цепочка) есть множество всех состояний, в которые из данного состояния возможен переход по данному входному символу или пустой цепочке (ε).

Принято полагать, что конечный автомат начинает работу в состоянии q_0, последовательно считывая по одному символу входного слова (цепочки входных символов). Считанный символ переводит автомат в новое состояние в соответствии с функцией переходов.

Читая входную цепочку символов x и делая переходы из состояния в состояние, автомат после прочтения последнего символа входного слова окажется в некотором состоянии q'.

Если это состояние является заключительным, то говорят, что автомат допустил слово x.

Конечные автоматы широко используются на практике, например, в синтаксических и лексических анализаторах, тестировании программного обеспечения на основе моделей.

Содержание

Другие способы описания

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.

Детерминированность

Конечные автоматы 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, помеченные пустой цепочкой ε, либо из одного состояния могут выходить несколько переходов, помеченных одним и тем же символом.

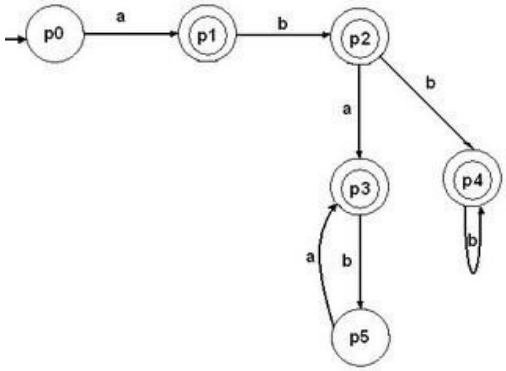


Рис.3
ДКА

Детерминированный конечный автомат

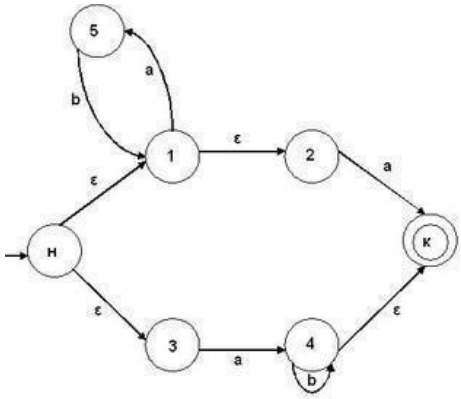
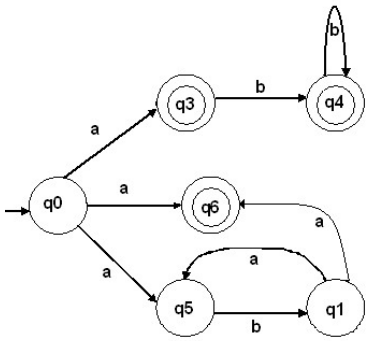


Рис.1
НКА с пустыми переходами
- конечное состояние

Недетерминированный автомат с пустыми переходами



Недетерминированный автомат с переходами из одного состояния помеченными одним и тем же символом

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

Специализированные языки программирования

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

В SFC программа описывается в виде схематической последовательности шагов, объединенных переходами.

Разработка моделей с использованием конечных автоматов

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 на конечном автомате приведет к быстрому росту числа состояний автомата, что в итоге сделает разработку такой модели крайне утомительным занятием. Как было отмечено выше, последнюю проблему можно решить, если использовать недетерминированный автомат.

Что может «делать» конечный автомат и последовательностная машина?

Ответ дается в различных терминах в зависимости от того, является ли автомат (соответственно П-машина) автономным или нет[1]. 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 определенный 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. *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.

См. также

- Автомат Мили
- Автомат Мура
- JFLAP — кроссплатформенная программа симулятор (автоматов, машины Тьюринга, грамматик)
- Автоматное программирование
- Sequential Function Chart
- Компилятор
- Транслятор
- Сети Петри
- Секвенциальная логика (Последовательностная логика)
- Таблица принятия решений

Литература

- *Белоусов А. И., Ткачев С. Б.* Дискретная математика. — М.: 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 (http://trpl7.ru/t-books/TRYAP_BOOK_Details.htm) — 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 (http://robot-develop.org/archives/1440#more-1440)
- Глушков В. М. Синтез цифровых автоматов. — М.: ГИФМЛ, 1962. — 476 с.

Ссылки

- *Дехтярь М. И.* Введение в схемы, автоматы и алгоритмы (http://www.intuit.ru/department/ds/introsaa/)
- Open source генератор конечных автоматов на языках C++ и Java по XML файлам описания (http://sourceforge.net/projects/genfsm/)
- Недетерминированные конечные автоматы (http://rsdn.org/article/alg/nka.xml)
- *Подзоров С. Ю.* Курс лекции по теории алгоритмов (http://www.nsu.ru/education/podzorov/Alg/Course.pdf)
