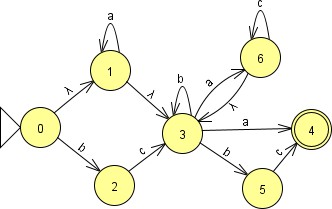
**Пример**. Построить ДКА, распознающий язык над алфавитом A={a,b,c}, заданный РВ: H=(a\*Vbc)(bVac\*)\*(aVbc). Скобки являются группировочными.

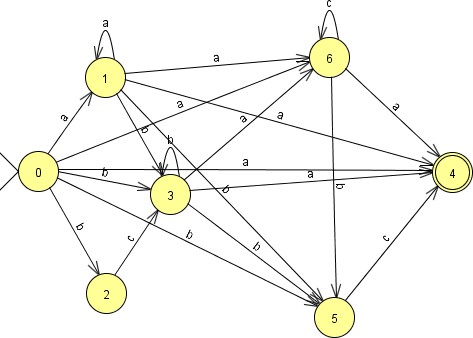
**Разбор решения**. **1.**В общем случае регулярное выражение, задающее событие, определяет *недетерминированный* автомат. Переход от РВ к ***εНКА*** - недетерминированному автомату, имеющему спонтанные переходы – задача сложности O(n). Важно при построении учитывать те 4 правила, которые приведены в лекции: они позволяют избежать избыточных путей и задать все необходимые пути в автомате, т.е. *точно* установить автомат, распознающий заданный событием язык. Если в РВ нет комбинации итераций/вложенных итераций, то спонтанных переходов в графе автомата может не появиться. Основное преимущество ***εНКА –*** наглядность представления за счет одного начального/одного конечного состояний и минимального нужного количества связей между состояниями.

1. От спонтанных переходов можно избавиться применением рекурсивной процедуры, связанной с изменением *связей* между состояниями. Так, для спонтанного перехода AB из A следует

«вытянуть» все дуги, аналогичные идущим из B с сохранением их маркировки, а после - оборвать дугу AB со спонтанным переходом. Если B-допускающее, то А – тоже допускающее; если А- начальное, то B- тоже начальное. Понять функционирование такого автомата существенно сложнее:

при неизменном количестве состояний изменяется их «роль/функция», возрастает *количество связей* между состояниями (диаграмма справа). Значительнее всего усложняет восприятие появившаяся множественность начальных состояний. Наглядность функционирования, присущая εНКА, исчезла напрочь.

Для получения ***единственного*** начального состояния можно ввести дополнительное состояние, сделать его начальным, соединить спонтанными переходами с 0-1- 3, а затем вновь выполнить процедуру устранения спонтанных переходов; состояния 0-1-3 при этом



станут переходящими. Важно, что если в связываемых состояниях есть допускающие, то новое начальное также будет допускающим. При наличии последовательных спонтанных дуг в автомате эта процедура ***«чувствительна»*** к ошибкам.

1. Во многих программных системах реализован другой алгоритмический подход, которых позволяет *непосредственно* по РВ построить НКА ***без*** спонтанных переходов и с единственным начальным состоянием, часто называемый технологией/алгоритмом разметки мест. Он связан с определением предосновных и основных мест в РВ (позиций до/после символа в анализируемом событии), раздаче этим местам индивидуальных индексов, соответствующих состоянию автомата. Разметка выполняется по правилам подчинения. Правила разметки приведены в методичке и будут подробно рассмотрены в лекции; наиболее частые ошибки разметки связаны с итерациями. В рассматриваемом примере разметка мест выглядит следующим образом:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **(** |  | a**\*** |  | **V** |  | **b** |  | **c** |  | **)** |  | **(** |  | **b** |  | **V** |  | **a** |  | **c\*** |  | **)\*** |  | **(** |  | **a** |  | **V** |  | **b** |  | **c** |  | **)** |  |
|  | 0 |  | 0 |  | 1 |  | 0 |  | 2 |  | 3 |  | 0 |  | 0 |  | 4 |  | 0 |  | 5 |  | 6 |  | 0 |  | 0 |  | 7 |  | 0 |  | 8 |  | 9 |  | 7 |
|  |  |  | 1 |  | 0 |  |  |  |  |  |  |  | 1 |  | 1 |  | |  | 1 |  | 6 |  | 5 |  | 1 |  | 1 |  |  |  | 1 |  |  |  |  |  | 9 |
|  |  |  |  |  |  |  |  |  |  |  |  |  | 3 |  | 3 |  |  |  | 3 |  |  |  |  |  | 3 |  | 3 |  |  |  | 3 |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 4 |  |  |  | 4 |  |  |  |  |  | 4 |  | 4 |  |  |  | 4 |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 6 |  |  |  | 6 |  |  |  |  |  | 6 |  | 6 |  |  |  | 6 |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 5 |  |  |  | 5 |  |  |  |  |  | 5 |  | 5 |  |  |  | 5 |  |  |  |  |  |  |

Очевидно, что индекс начального места при разметке – единственный, а индексов конечных мест (за ними стоят допускающие состояния автомата) может быть несколько. Для восстановления системы переходов в автомате нужно определить все предосновные и основные индексы, разделенные символом *входного* алфавита, т.к. группировочные скобки и символы операций не размечаются:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7\* | 8 | 9\* |
| a | 1|5|6|7 | 1|5|6|7 |  | 5|6|7 | 5|6|7 | 5|6|7 | 5|6|7 |  |  |  |
| b | 2|4|8 | 4|8 | 4 | 4|8 | 4|8 | 4|8 | 4|8 |  |  |  |
| с |  |  | 3 |  |  | 5|6 | 5|6 |  | 9 |  |

Недетерминированный автомат, представляющий заданный язык, построен. Нетрудно видеть, что состояния 5,6 в нем эквивалентны, равно как и 7/9, 3/4:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 5 | 7\* | 8 |
| a | 1|5|7 | 1|5|7 |  | 5|7 | 5|7 |  |  |
| b | 2|3|8 | 3|8 | 3 | 3|8 | 3|8 |  |  |
| с |  |  | 3 |  | 5 |  | 7 |

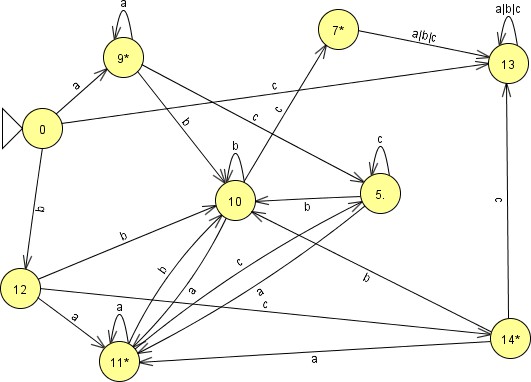
ВАЖНО! В отчетности допускается использование любой одной технологии (можно и двух - чтобы вам самим убедиться в правильности результата). Итоговым результатом должен быть **ДКА – структура** с **одним** начальным состоянием, определенностью и однозначностью переходов.

1. В данном примере начальное и допускающее состояния – единственные, т.к. событие начинается и заканчивается конкатенацией. Избавиться от множественных переходов можно рассматривая все множественные состояния НКА как новое состояние ДКА; дополнительно требуется введение состояния-стока для неопределенных переходов (см. пример по ссылке выше). В общем случае, в ДКА состояний может быть до 2n, где n- число состояний в НКА.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 5 | 7\* | 8 | 9\*=1|5|7\* | 10=3|8 | 11\*=5|7\* | 12=2|3|8 | 13=’ ’ | 14=3|7\* |
| a | 1|5|7 | 1|5|7 |  | 5|7 | 5|7 |  |  | 9 | 5|7 | 11 | 11 |  | 11 |
| b | 2|3|8 | 3|8 | 3 | 3|8 | 3|8 |  |  | 3|8 | 10 | 10 | 10 |  | 10 |
| с |  |  | 3 |  | 5 |  | 7 | 5 | 7 | 5 | 3|7 |  |  |

Поскольку в автомате задано начальное состояние 0, то отдельно взятые сами по себе состояния 1,2,3,8 недостижимы и их следует исключить (тримминг).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 5 | 7\* | 9\* | 10 | 11\* | 12 | 13 | 14\* |
| a | 9 | 11 | 13 | 9 | 11 | 11 | 11 | 13 | 11 |
| b | 12 | 10 | 13 | 10 | 10 | 10 | 10 | 13 | 10 |
| с | 13 | 5 | 13 | 5 | 7 | 5 | 14 | 13 | 13 |

Важно: несмотря на одинаковую систему переходов, состояния 7 и 13 – **не эквивалентны,** т.к. одно из них тупиковое, другое – допускающее; также не эквивалентны 5 и 11.

Такой автомат легко запрограммировать, он не требует сохранения копий, не совершает откатов при анализе цепочек и работает максимально быстро. Однако, восстановление события - как цели поиска (шаблона) - по ДКА является высокозатратной в вычислительном плане задачей.

1. проверка полученного решения H=(a\*Vbc)(bVac\*)\*(aVbc) на некоторых строках: Самые короткие допускаемые цепочки:
   1. а 2. аа 3.bc 4.bca 5. abc

не допускаемые (отвергаемые) цепочки:

1. все, которые переведут из 0 в 13, например, начинающиеся на «с»;
2. все, которые не доведут до 7|9|11|14, а оставят/переведут в промежуточные 5|10|12

Проверить решение можно в JFLAP. Следует правильно интерпретировать результаты этой

системы: маркированная знаком пустого множества «∅» дуга отражает пустое событие, т.е. соединяемые ею состояния не связаны.