КИБЕРНЕТИКА

С.Л. КРЫВЫЙ

УДК 51.681.3

КОНЕЧНЫЕ АВТОМАТЫ В ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЯХ

Ключевые слова: *конечный автомат, автомат Бюхи, автомат Мюллера, верификация.*

ВВЕДЕНИЕ

Теория автоматов, как один из разделов современной науки о вычислениях, возникла более 65 лет назад. К середине прошлого столетия в теории (конечных) автоматов были получены основные результаты: разрешимость проблем эквивалентности, синтеза, анализа, минимизации, детерминизации [1–5]. Установлен также класс языков, которые допускаются конечными автоматами (КА) [2, 6]. Языки этого класса получили название регулярных языков. Для них установлена разрешимость проблем пустоты, бесконечности, включения, замкнутости относительно теоретико-множественных операций и других проблем [1, 7].

В настоящее время теория конечных автоматов активно используется в системах обработки и поиска текстовой информации [8], верификации [4, 9], проектировании аппаратуры [10], биологии, генетике [8, 11, 12] и др. Популярность теории автоматов объясняется тем, что она имеет хорошую алгоритмическую основу в связи с разрешимостью перечисленных выше проблем для автоматов и языков. Благодаря этим свойствам конечных автоматов появились многие обобщения этого понятия: автоматы над бесконечными словами [13, 14], автоматы над деревьями [15–17], вероятностные и многоленточные автоматы [18, 19], временные автоматы [9, 20], клеточные автоматы [21], гибридные автоматы [22], магазинные автоматы [4, 6] и др.

В данной статье описываются некоторые области применения теории автоматов. Эти области различны по своей природе, поэтому изложение может показаться несколько разнородным, однако именно это, по мнению автора, и отражает разнообразие областей приложения теории автоматов. Здесь рассматривается применение конечных автоматов над словами конечной длины в области компьютерной алгебры, диофантовых ограничений, верификации свойств сетей Петри, биологии и в генетике. Кроме того, рассматривается применение конечных автоматов над словами бесконечной длины в области верификации свойств реактивных систем, которые специфицируются с помощью формул темпоральной логики, а также приложения временных автоматов к анализу свойств временных сетей Петри.

КОНЕЧНЫЕ АВТОМАТЫ НАД СЛОВАМИ КОНЕЧНОЙ ДЛИНЫ

Пусть $X = \{x_1, x_2, \ldots, x_n\}$ — конечный алфавит. Множество всех слов конечной длины в алфавите X будем обозначать X^*. Произвольное подмножество слов $L \subseteq X^*$ называется языком в алфавите X.

Конечным инициальным детерминированным X-автоматом называется пятерка $\mathcal{A} = (A, X, f, a_0, F)$, где A — конечное множество состояний, X — конечный алфавит, $f: A \times X \to A$ — функция переходов, $a_0 \in A$ — начальное состояние, $F \subseteq A$ — множество заключительных состояний. Алфавит X называется входным алфавитом автомата $\mathcal{A}$. Считается, что если $f(a, x) = b$, то автомат под действием входного символа $x \in X$ переходит из состояния a в состояние b, а если $f(a, p) = a'$, где $p \in X^*$, то слово p переводит автомат $\mathcal{A}$ из состояния a в состояние a'. Слово p допускается автоматом $\mathcal{A}$, если $f(a_0, p) \in F$. Язык L допускается автоматом $\mathcal{A}$, если все слова этого языка и только они допускаются этим автоматом. Языки, допускаемые конечными X-автоматами, называются регулярными.

1. КА в теории свободных полугрупп и групп. Если $P = \{p_1, p_2, \ldots, p_m\}$ — конечное множество слов из X^*, то слова $p_1, p_2, \ldots, p_m$ порождают полугруппу $L \subseteq X^*$ относительно операции конкатенации. Это значит, что $p \in L$ тогда и только тогда, когда слово p построено из слов $p_1, p_2, \ldots, p_m$ с помощью операции конкатенации. Поскольку множество слов L является регулярным языком ($L = \{p_1 \vee p_2 \vee \ldots \vee p_m\}$), то для него существует конечный X-автомат, который допускает слова этого языка и только их. Такой детерминированный автомат легко построить без применения общего алгоритма синтеза конечных автоматов. Тогда из теории конечных автоматов вытекает следующее утверждение.

Теорема 1. Пересечение конечного числа конечно-порожденных свободных полугрупп $L_1, L_2, \ldots, L_n$ в алфавите X является конечно-порожденной полугруппой. Существует алгоритм построения множества образующих полугруппы $L = L_1 \cap L_2 \cap \ldots \cap L_n$ по системам образующих полугрупп $L_1, L_2, \ldots, L_n$.

На этом свойстве конечно-порожденных полугрупп базируются способы идентификации слов и представления словарей естественных языков [8].

Аналогичные результаты имеют место и для свободных конечно-порожденных групп [23]. При этом вначале по системе образующих строится нильсеновский базис [24], а затем на основании этого базиса строится конечный детерминированный X-автомат, допускающий слова подгруппы, порожденной таким базисом. Имеет место следующая теорема.

Теорема 2. Пересечение конечного числа конечно-порожденных свободных групп $L_1, L_2, \ldots, L_n$ в алфавите X является конечно-порожденной группой. Существует алгоритм построения множества образующих группы $L = L_1 \cap L_2 \cap \ldots \cap L_n$ по нильсеновским базисам групп $L_1, L_2, \ldots, L_n$.

2. КА и линейные диофантовые уравнения. Одним из наиболее эффективных алгоритмов решения линейного однородного диофантового уравнения (ЛОДУ)

$$a_1 x_1 + a_2 x_2 + \ldots + a_q x_q = 0 \qquad (1)$$

является алгоритм Фортенбахера [25]. Этот алгоритм строит базис множества всех решений ЛОДУ, который состоит из минимальных решений относительно порядка $x = (x_1, x_2, \ldots, x_q) \le y = (y_1, y_2, \ldots, y_q) \Leftrightarrow x_i \le y_i$ для всех $i = 1, 2, \ldots, q$. Суть алгоритма Фортенбахера состоит в том, чтобы искать минимальные базисные решения ЛОДУ (1), начиная с канонических векторов N^q, т.е. с векторов $e_1 = (1, 0, \ldots, 0)$, $e_2 = (0, 1, \ldots, 0)$, ..., $e_q = (0, 0, \ldots, 1)$. При этом если некоторый текущий вектор $x = (x_1, \ldots, x_q)$ еще не является решением (1), то:

— при $a_1 x_1 + \ldots + a_q x_q < 0$ увеличить на единицу некоторое x_j такое, что $a_j > 0$;

— при $a_1x_1+\ldots+a_qx_q>0$ увеличить на единицу некоторое x_j такое, что $a_j<0$. (Заметим, что если все $a_i>0$ или все $a_i<0$, то уравнение (1) имеет лишь тривиальное решение $x_0=(0,\ldots,0)$.)

Сказанное можно выразить формально таким условием (условие Фортенбахера): увеличивать на единицу такое x_j, для которого $a(x)\cdot a(e_j)<0$, где $x=(x_1,\ldots,x_q)$, $a(e_j)=a_j$, а e_j — вектор канонического базиса.

Пример 1. Найти базис множества решений ЛОДУ $-x_1+x_2+2x_3-3x_4=0$.

Схема всего процесса решения показана на рис. 1. На векторах канонического базиса уравнение имеет значения $-1,\ 1,\ 2,-3$ (отражает верхний ряд схемы).

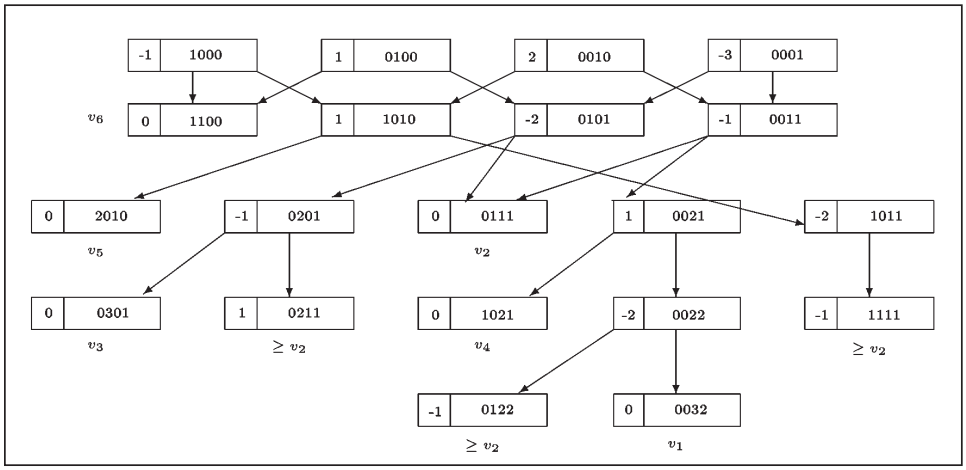

Рис. 1. Схема процесса решения ЛОДУ

В результате получаем такие базисные векторы:

$$v_1=(0,\ 0,\ 3,\ 2),\quad v_2=(0,\ 1,\ 1,\ 1),\quad v_3=(0,\ 3,\ 0,\ 1),$$
$$v_4=(1,\ 0,\ 2,\ 1),\quad v_5=(2,0,1,0),\qquad v_6=(1,\ 1,\ 0,\ 0).$$

Метод Фортенбахера можно переформулировать и обосновать с помощью частичных конечных X-автоматов. Существование и возможность построения соответствующего КА вытекает из условия Фортенбахера: сумма чисел $a(x)$ и $a(e_i)$, удовлетворяющих этому условию, всегда строго меньше по абсолютной величине абсолютных величин слагаемых. Это значит, что алгоритм Фортенбахера порождает процесс, который через конечное число шагов должен замкнуться. Формальное изложение этого подхода имеет следующий вид.

Пусть $a(x)=a_1x_1+a_2x_2+\ldots+a_qx_q=0$ является ЛОДУ. Входным алфавитом искомого автомата есть множество $X=\{e_1,e_2,\ldots,e_q\}$ векторов канонического базиса. Множество состояний Q автомата вначале включает единственное состояние 0, а остальные элементы этого множества порождаются в процессе построения X-автомата $\mathcal{A}=(Q,X,f,\{0\},\{0\})$ следующим образом:

$$f(m,e_i)=\begin{cases}a_i, & \text{если } m=0,\\ m+a(e_i), & \text{если } m\cdot a(e_i)<0,\\ *, & \text{если не определено,}\end{cases}$$

где $m\in Z$ — множество целых чисел, $e_i\in X$, $i=1,2,\ldots,q$, а символ * означает неопределенность.

После замыкания процесса генерации состояний автомата $\mathcal{A}$ строится базис множества решений уравнения $a(x)=0$. Для этого необходимо найти все слова p в алфавите X, соответствующие всем простым циклическим путям из состояния 0 в состояние 0, построить векторы-решения, соответствующие этим словам, и удалить повторяющиеся и неминимальные векторы (см. рис. 1).

Из свойств конечных автоматов и допускаемых ими языков следуют такие утверждения.

Теорема 3. Множество всех решений уравнения $a(x)=0$ является регулярным множеством. Базис множества решений этого уравнения состоит из всех слов в алфавите X, соответствующих всем различным простым циклическим путям из начального состояния 0 в состояние 0 автомата $\mathcal{A}$, представляющего множество решений данного уравнения [26].

Следствие 1. Алгоритм построения автомата $\mathcal{A}$ завершает свою работу после конечного числа шагов. Число состояний автомата $\mathcal{A}$, порождаемых в процессе его построения методом Фортенбахера, ограничено величиной $\sum_{i=1}^{q}|a_i|+1$.

Пример 2 (продолжение примера 1). Рассмотрим ЛОДУ $a(x)=-x_1+x_2+$ $+2x_3-3x_4=0$ из примера 1. Автомат, представляющий все решения данного уравнения, порождается следующим образом. Начальное значение множества $Q=0$. Состояние 0 является начальным и заключительным состоянием автомата. Генерируем новые состояния:

$$f(0,e_1)=-1,\ f(0,e_2)=1,\ f(0,e_3)=2,\ f(0,e_4)=-3.$$

Следовательно, множество $Q=\{0,1,2,-1,-3\}$. Вновь полученные состояния, в свою очередь, порождают такие состояния:

$$f(1,e_1)=0,\ f(-1,e_2)=0,\ f(-1,e_3)=1,\ f(2,e_1)=1,$$

$$f(1,e_4)=-2,\ f(2,e_4)=-1,\ f(-3,e_2)=-2,\ f(-3,e_3)=-1.$$

Новым есть состояние -2, которое порождает состояния $f(-2,e_2)=-1$, $f(-2,e_3)=0$. Поскольку эти состояния принадлежат множеству Q, то это значит, что процесс порождения состояний замкнулся. Следовательно, получаем автомат, таблица переходов которого имеет вид

f	0	-1	-2	1	2	-3
e_1	-1	*	*	0	1	*
e_2	1	0	-1	*	*	-2
e_3	2	1	0	*	*	-1
e_4	-3	*	*	-2	-1	*

Автоматный подход обобщается и на случай неоднородных ЛДУ, и на случай систем ЛДУ [25, 26].

3. КА и сети Петри. Применение КА к анализу свойств сетей Петри (СП) поясним на примере. Пусть имеется СП, которая моделирует работу насоса, перекачивающего жидкость из одной емкости p_1 в другую емкость p_3 (рис. 2). Место p_2 соответствует рабочему состоянию насоса, а p_4 — нерабочему состоянию насоса.

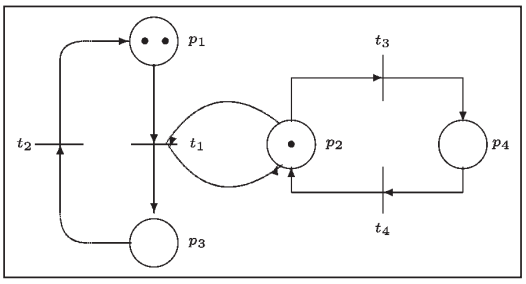

Рис. 2. Сеть Петри

Наиболее известным и практичным методом анализа свойств СП является уравнение состояния, представляемое системой линейных однородных диофантовых уравнений (СЛОДУ), решения которой находятся во множестве натуральных чисел N. Эта система имеет вид $A\cdot x=0$, где A — матрица инцидентности СП. Строки матрицы соответствуют местам СП,

а столбцы — переходам. Если в *i*-м столбце этой матрицы элементы имеют положительные знаки и принадлежат разным строкам, то это значит, что при срабатывании *i*-го перехода в эти места будет положено соответствующее количество фишек. Если элементы имеют отрицательные значения, то определенное количество фишек забирается из соответствующего места при срабатывании *i*-го перехода. Количество фишек, которое имеется в месте p_i, обозначим $M(p_i)$. В данном примере начальной разметкой СП является $M_0 = (2,1,0,0)$, а ее матрица инцидентности имеет вид

$$A = \begin{pmatrix} -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{pmatrix}.$$

Действительно, переход t_1 при срабатывании забирает фишку из мест p_1 и p_2 и помещает по одной фишке в места p_2 (возвращает фишку в то же место) и p_3. Поэтому в первом столбце первый элемент матрицы отрицательный (−1), а третий элемент положительный (+1). Остальные элементы равны нулю (второй элемент равен нулю, поскольку фишка забирается из места p_2 и туда возвращается). Решения СЛОДУ $A \cdot x = 0$ называются *T*-инвариантами, а решения СЛОДУ $A^T \cdot y = 0$ — *S*-инвариантами.

T-инварианты СП составляют минимальное порождающее множество решений СЛОДУ $A \cdot x = 0$. В данном примере такими решениями есть векторы

$$x_1 = (1,1,0,0), \quad x_2 = (0,0,1,1),$$

соответствующие двум последовательностям срабатываний переходов: $t_1 t_2$ и $t_3 t_4$. Эти инварианты свидетельствуют о том, что СП не имеет мертвых переходов, которые никогда не срабатывают в процессе функционирования СП.

S-инварианты СП составляют минимальное порождающее множество решений СЛОДУ $A^T \cdot y = 0$. В данном случае такими решениями есть векторы

$$y_1 = (1,0,1,0), \quad y_2 = (0,1,0,1),$$

что свидетельствует об ограниченности мест p_1, p_2, p_3, p_4 и соответственно ограниченности СП.

Если в процессе анализа свойств СП необходимо найти последовательность срабатываний ее переходов, которая приводит к определенному месту или описывает некоторое событие в СП, то по инвариантам СП это сделать практически невозможно. Отметим, что инварианты указывают только на переход, который сработал, но не указывают последовательности их срабатывания. Для этой цели служит транзиционная система, которая является графом достижимых разметок СП. Если СП ограничена, то граф достижимых разметок этой СП будет конечным автоматом. Граф *G* переходов такого автомата показан на рис. 3.

По этому автомату с помощью алгоритма анализа конечного *X*-автомата можно найти множество последовательностей срабатывания переходов, а также анализировать многие статические, динамические и логические свойства СП. Основными из них являются свойства взаимного исключения (*mutex*), справедливости (*fairness*) и живучести (*liveness*).

Согласно свойству *mutex* для данной СП в местах p_2 и p_4 не могут фишки находиться одновременно (т.е. насос не может находиться одновременно в рабочем и нерабочем состоянии). Это свойство следует из того, что в автомате все достижимые разметки удовлетворяют равенству $M(p_2) + M(p_4) = 1$.

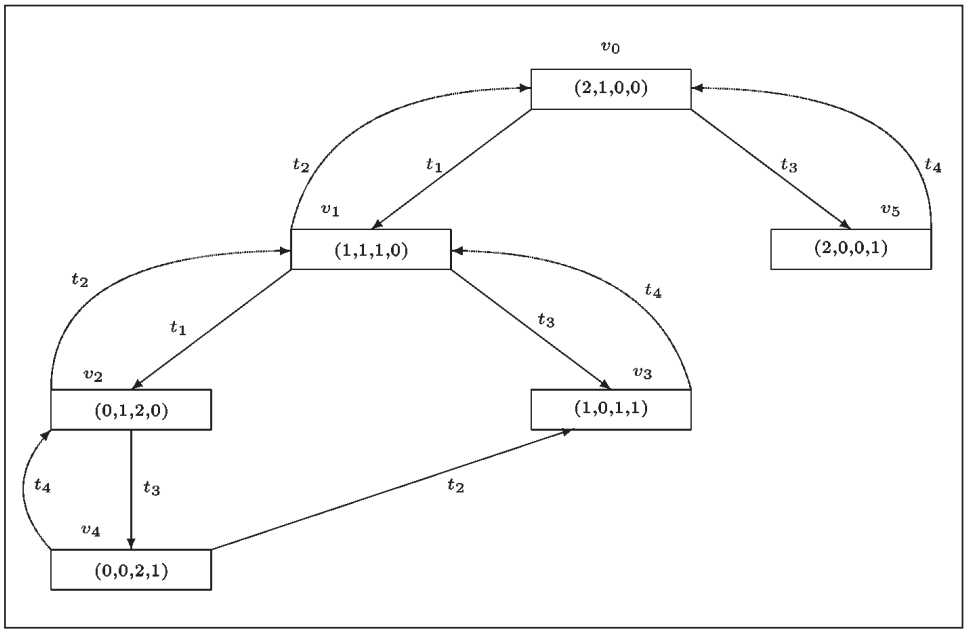

Рис. 3. Граф (автомат) G достижимых разметок СП

На основании свойства *fairness* для данной СП в месте p_4 в какой-то момент будет находиться фишка. Содержательно это свойство означает, что насос не будет работать постоянно, если наличие или отсутствие фишек в местах p_2 и p_4 интерпретировать как включение или отключение работы насоса, перекачивающего жидкость из первой емкости во вторую (из графа G видно, что формула $M(p_4)=1$ выполняется в вершинах v_3, v_4 и v_5).

Свойство *liveness* для данной СП состоит в том, что в ней при любой разметке всегда возможен хотя бы один переход. Это означает, что в СП нет разметки, при которой невозможен ни один переход. Содержательно свойство означает, что работа системы может продолжаться потенциально бесконечно при условии, что все контролирующие приборы будут работать правильно. Это свойство выполняется для данной СП, поскольку из каждого состояния в автомате G имеются переходы.

4. КА в биологии. Конечные автоматы в настоящее время успешно применяются в биологии. Создаются новые клетки и распознаются свойства существующих живых клеток, в частности распознаются свойства ДНК. Эти применения связаны со специальными языками, которые допускаются конечными автоматами [12].

Конструирование таких языков связано с получением новых слов из имеющихся. Неформально это выполняется следующим образом. Дана упорядоченная пара слов в алфавите a, b, c, d, например

$$u = ddddccaabbddd \text{ и } v = dddccaabbdddd .$$

По этой паре слов можно построить новое слово путем применения операции сечения каждого из них, например между двумя вхождениями символа a в подпоследовательности $s = ccaabb$, которая входит как в u, так и в v. После операции сечения применяется операция склеивания левой части первого слова и правой части второго слова. Первая операция, будучи применяемой к словам u и v, генерирует части $ddddcca$, $abbddd$ и $dddcca$, $abbdddd$. Применяя операцию склеивания к полученным частям, находим слово $x = ddddccaabbdddd$. Заметим, что, выполняя эти операции на упорядоченной паре v, u, получаем слово $y = dddccaabbddd$. В этом случае говорят, что слова x и y получены в результате применения операции сращивания пар слов u, v и v, u соответственно.

Сращивание слов происходит не произвольным способом, а в соответствии с определенными правилами, которые задаются изначально.

4.1. Правила сращивания, схемы, системы и языки. Рассмотрим формальные определения введенных выше понятий. Пусть $X = \{x_1, x_2, \ldots, x_n\}$ — конечный алфавит, X^* — множество всех слов конечной длины в алфавите X.

Правилом сращивания называется четверка $r = (u, u', v', v) \in (X^*)^4$. Действие правила r на словах языка определяет язык $r(L) = \{xuvy \in X^* \mid L$ содержит слова $xuu'q$ и $pv'vy$ для некоторых $x, p, q, y \in X^*\}$. Для множества правил сращивания R получаем язык $R(L) = \bigcup_{r \in R} r(L)$. Правило r относится к языку L, если язык $r(L)$ содержится в L. Множество правил R относится к языку L, если $R(L)$ содержится в L. Радиусом k правила сращивания $r = (u, u', v', v)$ называется максимум длин слов u, u', v', v.

Определение 1. Схемой сращивания называется пара $\sigma = (X, R)$, где X — конечный алфавит, R — конечное множество правил сращивания. Для каждого языка $L \subseteq X^*$ и натурального числа n определяется $\sigma^n(L)$:

$$\sigma^0(L) = L; \quad \sigma^{k+1}(L) = \sigma^k(L) \cup R(\sigma^k(L)).$$

Транзитивное замыкание последовательности $\sigma^0(L), \sigma^1(L), \sigma^2(L), \ldots$ $\ldots, \sigma^k(L), \ldots$ определяет язык $\sigma^*(L) = \bigcup_{n \geq 0} \sigma^n(L)$.

Системой сращивания называется пара (σ, I), где σ — схема сращивания, I — конечный начальный язык, содержащийся в X^*. Языком, порожденным системой (σ, I), называется язык $L(\sigma, I) = \sigma^*(I)$.

Язык L называется языком-сращиванием, если $L = L(\sigma, I)$ для некоторой системы сращивания (σ, I).

Пример 3. Пусть $X = \{a, b, c, d\}$, $r = (u, u', v', v)$, где слова $u, u', v', v \in X^*$ правила r такие, что $u = v' = cca$ и $u' = v = abb$. Пусть $R = \{r\}$, тогда схема сращивания имеет вид

$$\sigma = (X, R) = (\{a, b, c, d\}, \{cca, abb, cca, abb\}).$$

Пусть $I = \{ddddccaabbddd, dddccaabbdddd\}$. Применим правило r к упорядоченной паре слов $(ddddccaabbddd, dddccaabbdddd)$. Это правило порождает слово $dddccaabbdddd$, а с применением правила r к паре $(dddccaabbdddd, ddddccaabbddd)$ получаем слово $dddccaabbddd$.

Когда r действует на множестве $I \times I$, например на паре $(ddddccaabbddd, dddccaabbdddd)$, то результатом будет слово $ddddccaabbddd$, которое является координатой одной из пар. Отсюда получаем

$$\sigma^0(I) = I = \{ddddccaabbddd, dddccaabbdddd\})$$

и

$$\sigma^1(I) = \sigma^0(I) \cup R(\sigma^0(I)) =$$

$$= I \cup \{ddddccaabbdddd, dddccaabbddd, ddddccaabbddd, dddccaabbdddd\} =$$

$$= \{ddddccaabbdddd, dddccaabbddd, ddddccaabbddd, dddccaabbdddd\}.$$

Заметим, что R относится к языку $\sigma^1(I)$ и, значит, $\sigma^2(I) = \sigma^1(I)$. Тогда

$$\sigma^1(I) = \sigma^2(I) = \sigma^3(I) \text{ и } \sigma^*(I) = \sigma^1(I).$$

Отсюда следует, что $L(\sigma, I)$ — конечный язык.

Пример 4. Пусть $X = \{a, b, c, d\}$, $r = (b, bbbcc, b, bbbcc)$, $R = \{r\}$, $I = \{a^6 bbbbcca^6 bbbbcca^6\}$.

Правило r, применяемое в паре

$$(a^6 bbbbcca^6 bbbbcca^6,\ a^6 bbbbcca^6 bbbbcca^6)$$

к самому правому вхождению слова *bbbcc* правой координаты, порождает слово $a^6 bbbbcca^6 bbbbcca^6 bbbbcca^6$ длины 42.

Правило *r* можно применять и к левому вхождению слова *bbbbcc* в первой координате, и к правому вхождению этого слова во второй координате. Это порождает слово $a^6 bbbbcca^6$ длины 18. Следовательно,

$$\sigma^1(I) = \{a^6 bbbbcca^6, a^6 bbbbcca^6 bbbbcca^6, a^6 bbbbcca^6 bbbbcca^6 bbbbbcca^6\}.$$

Продолжая вычисления подобным образом, получаем бесконечный язык

$$L(\sigma, I) = a^6 bbbbcca^6 \{bbbbcca^6\}.$$

Символы слов вида $a^6 bbbbcca^6 bbbbcca^6$ в генетике интерпретируются как модель молекулы ДНК. Правило *r* представляет операции сечения и склеивания на некоторой субстанции в соединении ее с другой субстанцией. В этом понимании язык $L(\sigma, I) = a^6 bbbbbcca^6 \{bbbbcca^6\}$, полученный в примере 4, является моделью множества ДНК молекул, которые потенциально могут появляться при проверке на содержание некоторого вещества ДНК молекулы вида $a^6 bbbbcca^6 bbbbbcca^6$ или молекулы другого вещества.

Языки сращивания были введены в 1987 г. в работе [27], а позже в работах [28, 29] было показано, что языки сращивания являются регулярными языками, поскольку имеет место следующая теорема.

Теорема 4. Каждый язык сращивания является регулярным языком [28, 29].

В работе [30] показано, что не все регулярные языки являются языками сращивания. В [27] было доказано, что язык L является языком сращивания, если существует натуральное число n такое, что слово uxq принадлежит L, как только x имеет длину n и оба слова uxv и pxq принадлежат языку L.

Возникает вопрос: если дан регулярный язык L, то существует ли система сращивания, которая порождает язык L?

4.2. Синтаксические моноиды, соответствующие языкам сращивания. Для ответа на сформулированный выше вопрос необходимо показать, как решается следующая задача: будет ли данное правило сращивания относиться к регулярному языку $L \subseteq X^*$ [12].

Пусть $\mathcal{A} = (A, X, f, a_0, F)$ — минимальный детерминированный автомат, допускающий язык L. Для каждого состояния $a \in A$ и каждого слова $p \in X^*$ определим состояние $b = f(a, p)$, в которое переходит автомат $\mathcal{A}$ из состояния a под действием слова p. Заметим, что правило $r = (u, u', v', v)$ относится к языку L тогда и только тогда, когда для любой пары состояний $a, a' \in A$, для которой множества $\{x \in X^* \mid f(a, uu'x) \in F\}$ и $\{y \in X^* \mid f(a', v'vy) \in F\}$ непусты, множество $\{z \in X^* \mid f(a', v'vz) \in F\}$ включается в множество $\{z \in X^* \mid f(a, uvz) \in F\}$. Эти условия легко проверить, поскольку проблемы пустоты языка и включения для регулярных языков разрешимы.

Покажем теперь, каким образом специфицировать все правила, которые относятся к регулярному языку L, используя факт конечности числа правил. Эта спецификация опирается на отношение R синтаксической конгруэнтности, которое определяется на X^* и задается следующим образом: $p' Rp \Leftrightarrow$

$\Leftrightarrow (\forall x, y \in X^*)((xp'y \wedge xpy \in L) \vee (xp'y \wedge xpy \notin L))$. Фактор-множество X^*/R называется синтаксическим моноидом. В силу регулярности L число классов эквивалентности $n(L)$ по этому отношению будет конечным. Тогда существует в точности $m = (n(L))^4$ упорядоченных четверок классов конгруэнтности. Пусть (W, U, V, Z) — упорядоченная четверка классов конгруэнтности, $r = (w, x, y, z)$ и $r' = (w', x', y', z')$ — два правила сращивания из множества $W \times U \times V \times Z$. Докажем, что r относится к языку L тогда и только тогда, когда r' относится к языку L. В силу симметрии правил r и r' в этом утверждении достаточно показать, что если r относится к языку L, то r' также относится к языку L.

Пусть r относится к языку L и слова $uw'x'v$, $sy'z't \in L$, покажем, что $uw'z't \in L$. Из того, что $w'Rw$, получаем $uwx'v \in L$ и из того, что $x'Rx$, получаем $uwxv \in L$. Из того, что $y'Ry$ и $z'Rz$, следует, что $syzt \in L$. Поскольку r относится к L и $uwxv$, $syzt \in L$, то $uwzt \in L$, а из того, что $w'Rw$ и $z'Rz$, следует $uw'z't \in L$, что и требовалось показать.

Следовательно, для определения всех правил, которые относятся к языку L, необходимо построить $(n(L))^4$ четверок синтаксических классов на X^* с помощью L. Затем из каждой такой четверки (W, U, V, Z) следует выбрать по одному слову-представителю каждого класса для получения одного правила (w, x, y, z), чтобы определить, будет ли оно относиться к языку L. Если оно относится к L, то и каждое правило из $W \times U \times V \times Z$ относится к L. В противном случае ни одно правило из $W \times U \times V \times Z$ не относится к языку L. Таким образом, имеет место следующая теорема.

Теорема 5. Пусть L — регулярный язык. Множество правил, которые относятся к языку L, имеет вид

$$\bigcup_{i=1}^{m} \{W_i \times U_i \times V_i \times Z_i\},$$

где $m = (n(L))^4 > 0$ — натуральное число и каждое из множеств W_i, U_i, V_i, Z_i $(1 \le i \le m)$ является элементом синтаксического моноида языка L.

Поскольку каждый синтаксический класс регулярного языка сам является регулярным языком, то перечислим все слова длины, не большей радиуса k правил сращивания в этом классе. Если представление, данное в теореме 5, построено, то множество всех правил радиуса, не большего k, сохраняющих язык L, можно перечислить без дополнительных проверок. Для каждого множества $W_i \times U_i \times V_i \times Z_i$ $(1 \le i \le m)$ в этом представлении перечислим все правила (w, x, y, z) из $W_i \times U_i \times V_i \times Z_i$ радиуса, не большего k. Для того чтобы выполнить такое перечисление без использования синтаксического моноида, необходимо перебрать каждое из правил радиуса, не большего k, в множестве $(n(L))^4$ и проверить каждое из них на предмет сохранения языка L [31].

КОНЕЧНЫЕ АВТОМАТЫ НАД СЛОВАМИ БЕСКОНЕЧНОЙ ДЛИНЫ

Обобщением конечных X-автоматов являются КА над бесконечными словами. В основе такого обобщения лежит модель X-автомата, аналогичная рассмотренной выше, однако на вход данного автомата подаются слова не конечной, а бесконечной длины. В связи с этим условия распознавания языка, воспринимаемого таким автоматом, изменяются. При этом изменяются и свойства таких автоматов. Имеется два типа автоматов этого рода: автоматы Бюхи и обобщенные автоматы Бюхи, называемые автоматами Мюллера [14].

1. Основные свойства. Пусть $X = \{x_1, x_2, \ldots, x_n\}$ — конечный алфавит. Множество всех слов бесконечной длины в этом алфавите обозначим X^ω, а произвольное подмножество L множества X^ω будем называть ω-языком.

Определение 2. Недетерминированным автоматом Бюхи $\mathcal{A}$ над словами бесконечной длины в алфавите X называется пятерка (A, X, f, A_0, F), где A — конечное множество состояний автомата, $A_0 \subset A$ — множество начальных состояний, $f \subset A \times X \times A$ — отношение переходов, $F \subseteq A$ — множество заключительных состояний.

Недетерминированным автоматом Мюллера $\mathcal{A}$ над словами бесконечной длины в алфавите X называется пятерка $(A, X, f, A_0, \mathcal{F})$, где A, f, A_0 обозначены выше, а множество заключительных состояний $\mathcal{F} \subseteq B(A)$, где $B(A)$ — булеан множества A.

Четверка (A, X, f, A_0) называется таблицей переходов автомата.

Автоматы Бюхи и Мюллера часто называют ω-автоматами, если нет необходимости их различать. Таблица переходов (A, X, f, A_0) называется детерминированной, если $|A_0| = 1$, и для любых $x \in X$ и $a \in A$ существует не более одного $a' \in A$ такого, что $(a, x, a') \in f$. Называется ω-автомат детерминированным, если детерминирована его таблица переходов; ω-автомат стартует в одном из начальных состояний; если $(s, x, s') \in f$, то автомат может изменить свое состояние s на s' под действием входного символа $x \in X$.

Для слова $\sigma = x_1 x_2 x_3 \ldots$ в алфавите X выражение

$$r = s_0 \xrightarrow{x_1} s_1 \xrightarrow{x_2} s_2 \xrightarrow{x_3} \ldots$$

называется трассой автомата $\mathcal{A}$ над σ, которая начинается в состоянии $s_0 \in A_0$ и $(s_{i-1}, x_i, s_i) \in f$ для всех $i \geq 1$. Для трассы r множество $\inf(r)$ состоит из состояний $s \in A$ таких, что $s = s_i$ для бесконечного числа раз $i \geq 0$.

Трасса r автомата $\mathcal{A}$ над словом $\sigma \in X^\omega$ называется воспринимающей трассой тогда и только тогда, когда $\inf(r) \cap F \neq \varnothing$ $(\inf(r) \in \mathcal{F})$. Другими словами, трасса r является воспринимающей тогда и только тогда, когда некоторое состояние из множества F (множество из $\mathcal{F}$) повторяется бесконечно часто на трассе r. Язык $L \subseteq X^\omega$ называется допускаемым автоматом $\mathcal{A}$, если он состоит из слов $\sigma \in X^\omega$ таких, что автомат $\mathcal{A}$ имеет трассу r, которая воспринимает σ. В этом случае язык L будем обозначать $L(\mathcal{A})$.

Определение 3. ω-язык называется регулярным ω-языком тогда и только тогда, когда он допускается некоторым автоматом Бюхи.

Теорема 6. Приведем основные свойства недетерминированных автоматов Бюхи [13, 14]:

а) ω-язык $L \subseteq X^\omega$ допускается некоторым автоматом Бюхи тогда и только тогда, когда этот язык является объединением конечного числа множеств вида $U \cdot V^\omega$, где $U, V \subseteq X^*$ являются регулярными языками слов конечной длины; более того, справедливо включение $V \cdot V \subseteq V$;

б) если $V \subseteq X^*$ является регулярным языком, то язык V^ω допускается некоторым автоматом Бюхи;

в) если $V \subseteq X^*$ — регулярный язык и $L \subseteq X^\omega$ является языком, который допускается автоматом Бюхи, то язык $V \cdot L$ является языком, который также допускается некоторым автоматом Бюхи;

г) класс языков, допускаемых недетерминированными автоматами Бюхи, замкнут относительно операций объединения, пересечения и дополнения;

д) ω-регулярный язык непуст тогда и только тогда, когда он включает бесконечное ω-слово периодического характера $pqqq\cdots$;

е) проблема пустоты языка для автоматов Бюхи является алгоритмически разрешимой.

На этих свойствах автоматов построено большинство алгоритмов проверки выполнимости формул на моделях (алгоритмы *model checking*) в модальных логиках.

Отметим, что классы языков, воспринимаемых недетерминированными автоматами Бюхи, а также детерминированными и недетерминированными автоматами Мюллера, совпадают. Класс детерминированных автоматов Бюхи не замкнут относительно операции дополнения языков, поэтому он является собственным подклассом класса недетерминированных автоматов Бюхи.

2. КА, линейная логика и верификация. КА над бесконечными словами используются в процедурах моделирования и верификации свойств реактивных систем. В процессе такой верификации используются модель Крипке, которая моделирует верифицируемую систему, и логическая формула, описывающая ожидаемое поведение верифицируемой системы, выполнимость которой проверяется на этой модели (*model checking*). Логическая формула называется спецификацией системы, и для ее записи часто используется язык линейной темпоральной логики (ЛТЛ). Такая верификация возможна благодаря транслируемости в ω-автоматы как модели Крипке, так и логической формулы.

Существует несколько алгоритмов трансляции ЛТЛ-формул в ω-автоматы. Большинство таких алгоритмов базируется на методе семантического табло [32]. Наиболее популярным является алгоритм, предложенный в монографии [33]. Этот алгоритм по ЛТЛ-формуле φ строит соответствующий автомат Бюхи $\mathcal{A}_\varphi$ с числом состояний, которое не превышает величины $2^{O(|\varphi|)}$, где $|\varphi|$ — длина формулы φ. Построенный автомат допускает язык $L = \{\omega \in X^\omega \mid \omega \models \varphi\}$, где $X = B(AP)$ — булеан множества AP атомарных формул формулы φ.

Чтобы применить этот алгоритм, необходимо ЛТЛ-формулу привести к так называемой негативной нормальной форме, в которой отрицания применяются только к пропозициональным переменным.

Алгоритм трансляции ЛТЛ-формул в ω-автоматы в рамках ограниченного объема статьи изложить затруднительно, поэтому здесь будут приведены только некоторые подготовительные части этого алгоритма. (Более подробно этот вопрос изложен в [33].) Рассмотрим сначала способ трансляции моделей Крипке в ω-автоматы.

Трансляция моделей Крипке. Моделью Крипке над алфавитом атомарных формул AP называется четверка $M = (S, R, S_0, f)$, где S — конечное множество, элементы которого называются точками; $R \subseteq S \times S$ — бинарное отношение переходов между точками; $S_0 \subseteq S$ — множество начальных точек; $f: S \to B(AP)$ — функция, сопоставляющая данной точке $s \in S$ множество атомарных формул, истинных в этой точке.

Для моделей Крипке проблем с трансляцией в ω-автоматы не возникает, поскольку она непосредственно соотносится с ω-автоматом (Бюхи или Мюллера), все состояния которого заключительные, а множество возможных поведений системы задается ω-языком $L(\mathcal{A})$, допускаемым соответствующим автоматом $\mathcal{A}$.

Формальное определение трансляции модели Крипке в ω-автомат следующее.

Пусть $M = (S, R, S_0, f)$ — модель Крипке над множеством атомарных формул AP. По этой модели строим автомат Бюхи $\mathcal{A} = (A, X, Q, \{a_0\}, F)$, где $A = S \cup \{a_0\}$ — множество состояний; a_0 — единственное начальное состояние; $X = B(AP)$, где $B(AP)$ — булеан множества AP; $F = A$ — множество заключительных состояний; Q — отношение переходов, которое определяется таким образом: $\forall a, a' \in A \wedge x \in X$ тройка $(a, x, a') \in Q$ тогда и только тогда, когда $f(a) = x \wedge ((a, a') \in R \vee (a = a_0 \wedge a' \in S_0))$.

Пример 5. Пусть дана модель Крипке (рис. 4), в которой $s_1, s_2 \in S_0$, $S = \{s_1, s_2, s_3\}$. Тогда такой модели соответствует автомат Бюхи, представленный на рис. 5.

Спецификацию φ также можно представить в виде автомата $\mathcal{A}_\varphi$ над тем же алфавитом $X = B(AP)$. В этом случае ω-язык $L(\mathcal{A}_\varphi)$ определяет множество допустимых поведений системы.

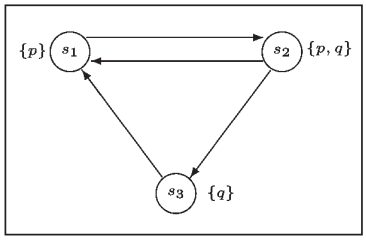

Рис. 4. Модель Крипке

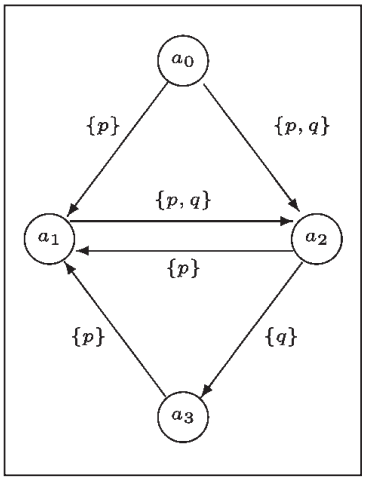

Рис. 5. Автомат Бюхи для модели Крипке

Трансляция ЛТЛ-формул в ω-автоматы. Алгоритм трансляции требует определения множества подформул для данной ЛТЛ-формулы φ. Это множество, обозначаемое $sub(\varphi)$, определяется следующим образом:

— булевы константы 0, 1 и сама формула φ принадлежат $sub(\varphi)$;

— если $\varphi_1 \vee \varphi_2 \in sub(\varphi)$, то $\{\varphi_1, \varphi_2\} \subseteq sub(\varphi)$;

— если $\varphi_1 \wedge \varphi_2 \in sub(\varphi)$, то $\{\varphi_1, \varphi_2\} \subseteq sub(\varphi)$;

— если $\bigcirc \varphi_1 \in sub(\varphi)$, то $\{\varphi_1\} \subseteq sub(\varphi)$;

— если $\varphi_1 \mathbf{U} \varphi_2 \in sub(\varphi)$, то $\{\varphi_1, \varphi_2\} \subseteq sub(\varphi)$;

— если $\varphi_1 \mathbf{R} \varphi_2 \in sub(\varphi)$, то $\{\varphi_1, \varphi_2\} \subseteq sub(\varphi)$,

где $\bigcirc$, $\mathbf{U}$, $\mathbf{R}$ — операторы ЛТЛ, называемые *next-time*, *untill* и *release* соответственно. Очевидно, что $|sub(\varphi)| = O(|\varphi|)$, а число подмножеств множества $sub(\varphi)$ имеет вид $|B(sub(\varphi))| = 2^{O(|\varphi|)}$, где $|\varphi|$ — длина формулы φ.

Основная идея алгоритма трансляции основывается на таких свойствах семантики ЛТЛ для $w \in S$:

— для доказательства того, что $w \models \varphi_1 \vee \varphi_2$, необходимо доказать выполнимость $w \models \varphi_1$ или $w \models \varphi_2$;

— для доказательства того, что $w \models \varphi_1 \mathbf{U} \varphi_2$, необходимо доказать выполнимость $w \models \varphi_1$ или $(w \models \varphi_2$ и $w \models \bigcirc(\varphi_1 \mathbf{U} \varphi_2))$;

— для доказательства того, что $w \models \varphi_1 \mathbf{R} \varphi_2$, необходимо доказать выполнимость $w \models \varphi_1$ и $w \models \varphi_2$ или $(w \models \varphi_2$ и $w \models \bigcirc(\varphi_1 \mathbf{R} \varphi_2))$.

Алгоритм трансляции в процессе работы строит граф переходов автомата, допускающий все интерпретации, при которых выполняется данная формула.

ВРЕМЕННЫЕ КОНЕЧНЫЕ АВТОМАТЫ НАД СЛОВАМИ БЕСКОНЕЧНОЙ ДЛИНЫ

В качестве модели временного автомата (ВА) выбираются автоматы Бюхи и Мюллера, на состояния и переходы которых накладываются временные ограничения. Свойства ВА, используемые в системах верификации, в основном те же, что и для автоматов Бюхи и Мюллера.

1. Основные свойства временных КА. Приведем сначала неформальное определение ВА. Временной автомат представляет собой конечный ω-автомат, снабженный часами, значения которых принадлежат множеству рациональных неотрицательных чисел Q^+. Считается, что переходы ВА из одного состояния в другое происходят мгновенно, но течение времени не останавливается, пока автомат находится в некотором состоянии. При выполнении перехода значения некоторых часов могут сбрасываться в нуль. В каждый момент времени значения часов фиксируют время, которое прошло с момента последнего сбрасывания часов. Считается, что темп времени одинаковый для всех часов, причем конечное число переходов может быть выполнено на конечном отрезке времени.

С каждым переходом связано некоторое временное ограничение. Переход может произойти только тогда, когда текущие значения часов удовлетворяют временному ограничению данного перехода. Временные ограничения также связаны с каждым состоянием автомата, эти ограничения называются инвариантами состояний. Время в состоянии течет до тех пор, пока инвариант этого состояния истинный.

Пример 6. Рассмотрим временной автомат, представленный на рис. 6. Начальным состоянием в этом автомате является s_0 . Автомат может находиться в этом состоянии до тех пор, пока значение часов y не превысит пяти. Как только значение часов y станет бóльшим или равным трем, автомат может выполнить переход под действием символа a в состояние s_1 и сбросить значение часов x в нуль.

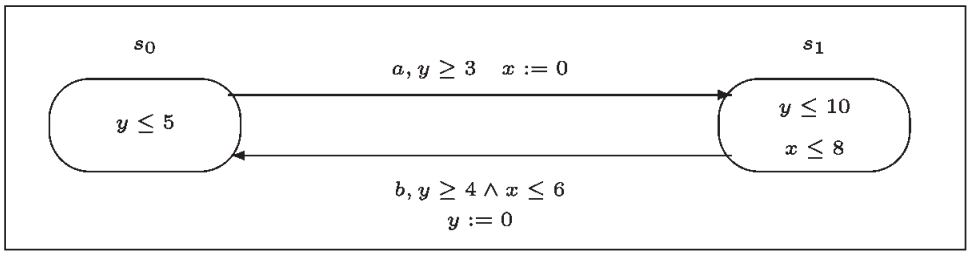

Рис. 6. Временной автомат

Автомат может оставаться в состоянии s_1 до тех пор, пока y не станет больше десяти и x не станет больше восьми. Когда значение часов y достигнет значения четырех, а значение часов x будет не больше шести, автомат получает возможность перейти под действием символа b в состояние s_0 и сбросить значение часов y в нуль.

Остается объяснить поведение ВА в состоянии. Предполагается, что течение времени бесконечно, т.е. в каждом состоянии временные ограничения сверху, которые накладываются на значения часов, или равны бесконечности, или меньше максимального предела, который определяется инвариантом состояния и условиями переходов, возможных для ВА. Иными словами, в каждом состоянии ВА может оставаться бесконечно долго или инвариант состояния заставит покинуть это состояние и в этот момент в ВА возможен хотя бы один переход.

Временные ограничения определяются индуктивно следующим образом.

Определение 4. Для множества переменных-часов C множество временных ограничений E включает ограничения вида $\delta = true \,|\, x \leq c \,|\, c \leq x \,|\, \neg\delta \,|\, \delta_1 \wedge \delta_2$, где $x \in C$, а $c \in Q^+$ — рациональное неотрицательное число.

Определение 5. Конечным недетерминированным временным автоматом Бюхи (ВАБ) над алфавитом X называется упорядоченная шестерка $\mathcal{A} = (A, X, A_0, C, E, F)$, где A — конечное множество состояний; $A_0 \subseteq A$ — множество начальных состояний; C — множество часов; $E \subseteq A \times X \times A \times \times B(C) \times \Phi(C)$ — множество временных ограничений ($B(C)$ — булеан множества C , а $\Phi(C)$ — множество формул над C); $F \subseteq A$ — множество заключительных состояний.

Если в этом определении вместо множества $F \subseteq A$ рассматривать семейство множеств $\mathcal{F} \subseteq B(A)$, где $B(A)$ — булеан множества A , то такой автомат называется конечным временным автоматом Мюллера (ВАМ).

Если переход $(a, \sigma, a', \lambda, \delta) \in E$, то происходит переход из состояния a в состояние a' под действием символа $\sigma \in X$, при котором часы из множества $\lambda \subseteq C$ сбрасываются в нуль, а $\delta \in \Phi(C)$ — временные ограничения этого перехода.

ВА начинает свою работу в одном из начальных состояний, когда все часы имеют нулевые значения. В момент времени τ_i автомат осуществляет переход из состояния a в состояние a', читая входной символ $\sigma \in X$ и используя некоторый переход $(a, \sigma, a', \lambda, \delta) \in E$, если текущие значения часов удовлетворяют временным ограничениям $\delta \in \Phi(C)$. Все значения часов из множества λ сбрасываются в нуль. Следовательно, каждому входному слову $p \in X^\omega$ соответствует последовательность τ , элементами которой являются значения часов, соответствующие моментам времени чтения символов слова p . Пару (p, τ) называют временным словом ВА.

Трассой r ВА над временным словом (p, τ) называется бесконечная последовательность вида

$$r: (s_0, v_0)\frac{\sigma_1}{\tau_1}(s_1, v_1)\frac{\sigma_2}{\tau_2}(s_2, v_2)\frac{\sigma_3}{\tau_3}\dots,$$

где $s_i \in A$, $\sigma_i \in X$, а $v_i \in [C \to R]$ для $i \geq 1$ удовлетворяют условиям:

— $s_0 \in A_0$ и $v_0(x) = 0$ для всех $x \in C$;

— для всех $i \geq 1$ в множестве E существует переход $(s_{i-1}, \sigma_i, s_i, \lambda_i, \delta_i)$ такой, что $(v_{i-1} + \tau_i - \tau_{i-1})$ удовлетворяет δ_i, а v_i имеет вид $[\lambda_i \to 0](v_{i-1} + \tau_i - \tau_{i-1})$.

Множество $\inf(r)$ состоит из таких состояний s, что $s = s_i \in F$ $(F = F_i \in \mathcal{F})$ появляется бесконечное число раз $i \geq 0$.

Выполнение run $r = (\bar{s}, \bar{v})$ ВАБ $\mathcal{A}$ над временным словом (σ, τ) допускается ВА $\mathcal{A}$ тогда и только тогда, когда $\inf(r) \cap F \neq \varnothing$ $(\inf(r) \in \mathcal{F})$. Язык $L(\mathcal{A})$ временных слов, которые допускаются ВА $\mathcal{A}$, определяется как множество всех временных слов (σ, τ) таких, что выполнение r над (σ, τ) допускается автоматом $\mathcal{A}$. Временной язык L называется временным регулярным языком тогда и только тогда, когда существует такой ВА $\mathcal{A}$, что $L = L(\mathcal{A})$. Для ВАБ и ВАМ имеют место утверждения, аналогичные утверждению для обычных автоматов Бюхи и Мюллера [9].

Теорема 7. Класс временных языков, допускаемых детерминированными ВАБ, замкнут относительно (конечных) пересечений и объединений и не замкнут относительно дополнения.

Теорема 8. Класс временных языков, которые допускаются детерминированными ВАМ, совпадает с классом языков, допускаемых недетерминированными ВАБ и ВАМ, и этот класс замкнут относительно операций объединения, пересечения и дополнения.

Отметим, что правила функционирования ВА могут иметь различную семантику, некоторые из них описаны в [9, 34].

2. КА и временные сети Петри. Временная СП — это обычная сеть Петри, срабатывание переходов которой должно удовлетворять определенным временным ограничениям. Формальное определение дано в [34].

Определение 6. Временной СП (ВСП) называется упорядоченная шестерка $\mathcal{N} = (P, T, F, M_0, Et, Lt)$, где (P, T, F, M_0) — сеть Петри и $Et: T \to N$, $Lt: T \to N \cup \{\infty\}$ — функции, определяющие временные интервалы возможного срабатывания переходов СП, причем $\forall t \in T(Et(t) \leq Lt(t))$.

Пусть $en(M)$ — множество переходов, которые могут сработать при разметке M. Это множество может отличаться от всего множества переходов T, но при исследовании поведения ВСП достаточно иметь только это множество.

Пример 7. Рассмотрим ВСП, изображенную на рис. 7, где для $i = 1, 3, 5, 6$ имеем $Et(t_i) = 1$, $Lt(t_i) = 2$, $Et(t_2) = 0$, $Lt(t_2) = 3$, $Et(t_4) = Lt(t_4) = 1$, $M_0 = (1, 1, 0, 0, 0, 0, 0, 0)$, $en(M_0) = t_1, t_2$.

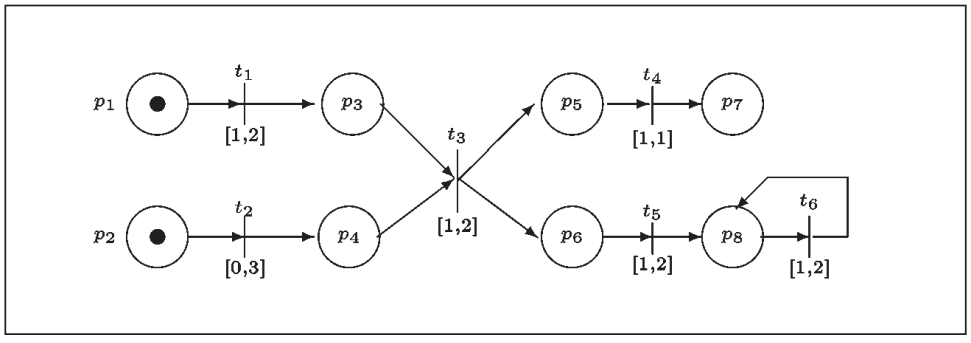

Рис. 7. Временная СП

Время, ассоциированное с переходами, изменяется таким образом. Назовем конкретным состоянием ВСП $\mathcal{N}$ пару

$$\sigma^T = (M, clock^T),$$

где M — разметка ВСП $\mathcal{N}$, а $clock^T : T \to Q^+$ — функция, сопоставляющая каждому возможному переходу $t \in T$ при разметке M $(t \in en(M))$ время.

Для $\delta \in Q^+$ и $t \in T$ выражение $clock^T + \delta$ означает время $clock^T(t) + \delta$. Далее будем считать, что $(M, clock^T) + \delta = (M, clock^T + \delta)$.

Пространством конкретных состояний ВСП $\mathcal{N}$ называется транзиционная система

$$C_c^T(\mathcal{N}) = (CP^T, (\sigma^T)^0, \to_{T_c}).$$

Здесь CP^T — множество конкретных состояний ВСП $\mathcal{N}$; $(\sigma^T)^0 = (M_0, clock_0^T)$, где $\forall t \in T(clock_0^T = 0)$ при начальной разметке M_0; $\to_{T_c}$ — отношение переходов, где:

— для $\delta \in Q^+ (M, clock^T) \overset{\delta}{\to}_{T_c} (M, clock^T + \delta) \Leftrightarrow (clock^T + \delta)(t) \le Lt(t)$ $\forall t \in en(M)$ (наследник времени);

— для $t \in T$ $(M, clock^T) \overset{t}{\to}_{T_c} (M_1, clock_1^T) \Leftrightarrow t \in en(M), Et(t) \le clock^T(t) \le Lt(t)$, $M \overset{t}{\to} M_1$ и $\forall u \in T(clock_1^T(u) = 0)$, если $u \in newly\text{-}en(M, t)$, иначе $clock_1^T = clock^T(u)$ (наследник действия) (*newly-en*(M, t) — совокупность переходов, срабатывания которых возможны после выполнения перехода t при разметке M).

Время, ассоциированное с местами, изменяется аналогичным образом. Пусть пара $\sigma^P = (M, clock^P)$ означает конкретное состояние ВСП $\mathcal{N}$, где M — разметка ВСП $\mathcal{N}$, $clock^P : P \to Q^+$ — функция, сопоставляющая каждому месту $p \in P$ при разметке M время. Для $\delta \in Q^+$ и $p \in P$ выражение $clock^P + \delta$ означает время $clock^P(p) + \delta$. Далее будем считать, что $(M, clock^P) + \delta = (M, clock^P + \delta)$.

Пространством конкретных состояний ВСП $\mathcal{N}$ называется транзиционная система

$$C_c^P(\mathcal{N}) = (CP^P, (\sigma^P)^0, \to_{P_c}).$$

Здесь CP^P — множество конкретных состояний ВСП $\mathcal{N}$; $(\sigma^P)^0 = (M_0, clock_0^P)$, где $\forall p \in P(clock_0^P = 0)$ при начальной разметке M_0; $\to_{P_c}$ — отношение переходов, где:

— для $\delta \in Q^+ (M, clock^P) \overset{\delta}{\to}_{P_c} (M, clock^P + \delta) \Leftrightarrow$ для каждого $t \in en(M)$ существует место $p \in {}^\bullet t$ такое, что $(clock^P + \delta)(p) \le Lt(t)$ (наследник времени);

— для $t \in T$ $(M, clock^P) \overset{t}{\to}_{P_c} (M_1, clock_1^P) \Leftrightarrow t \in en(M)$, для каждого $p \in {}^\bullet t$ имеет место $Et(t) \le clock^P(p)$, существует $p \in {}^\bullet t$ такое, что $clock^P(p) \le Lt(t)$, $M \overset{t}{\to} M_1$ и $\forall p \in P(clock_1^P(p) = 0)$, если $p \in {}^\bullet t$, иначе $clock_1^P(p) = clock^P(p)$ (наследник действия).

Трансляция ВСП в ВА происходит сопоставлением множествам конкретных состояний временных ограничений.

Пусть $\mathcal{N} = (P, T, F, M_0, Et, Lt)$ — временная сеть Петри. Предполагается, что ВСП ограниченная и ординарная. Такой ВСП сопоставляется ВА $A_{\mathcal{N}} = (A_{\mathcal{N}}, L_{\mathcal{N}}, A_0, C_{\mathcal{N}}, E_{\mathcal{N}}, F_{\mathcal{N}})$. В этом автомате множеством состояний выступает множество всех достижимых разметок, символами входного алфавита являются символы множества переходов T, $A_0 = M_0$. Множество значений часов имеет вид $C_{\mathcal{N}} = \{x_t \mid t \in T\}$, т.е. часы определены для каждого перехода ВСП $\mathcal{N}$. Инварианты состояний $M \in A_{\mathcal{N}}$ находятся следующим образом:

$$F_{\mathcal{N}}(M) = \begin{cases} \bigvee\limits_{\{t \in T \mid t \in en(M) \wedge Lt(t) < \infty\}} x_t \le Lt(t), \text{ если } \{t \in T \mid t \in en(M) \wedge Lt(t) < \infty \ne \varnothing\}, \\ \text{true}, \quad \text{если иначе.} \end{cases}$$

Множество переходов определяется так:

$$E_{\mathcal{N}} = \{e_{M,t} \mid t \in en(M) \wedge M \in A_{\mathcal{N}}\}.$$

Для заданного перехода $e_{M,t} \in E_{\mathcal{N}}$, соответствующего срабатывающему переходу $t \in en(M)$ при разметке M, условие перехода находится из выражения

$$guard_{\mathcal{N}}(e_{M,t}) = (x_t \ge Et(t)),$$

а множество значений часов устанавливается равенством

$$reset_{\mathcal{N}}(M, t) = \{x_u \in C_{\mathcal{N}} \mid x_u \in newly\text{-}en(M, t)\}.$$

Переходы автомата $A_{\mathcal{N}}$ имеют вид

$$e_{M,t} := M \xrightarrow{t, cc, X} M_1,$$

где $cc = guard_{\mathcal{N}}(e_{M,t})$ и $X = reset_{\mathcal{N}}(M, t)$.

Пропозициональные переменные PV_P определяются в зависимости от места ВСП, а функция значений $V_{\mathcal{A}_{\mathcal{N}}} : A_{\mathcal{N}} \to B(PV_P)$ имеет вид

$$V_{\mathcal{A}_{\mathcal{N}}}(M) = \bigcup_{\{p \in P \mid M(p) > 0\}} V_{\mathcal{N}}(p).$$

При этом две модели ВСП и соответствующего ей ВА должны быть бисимуляционно эквивалентны.

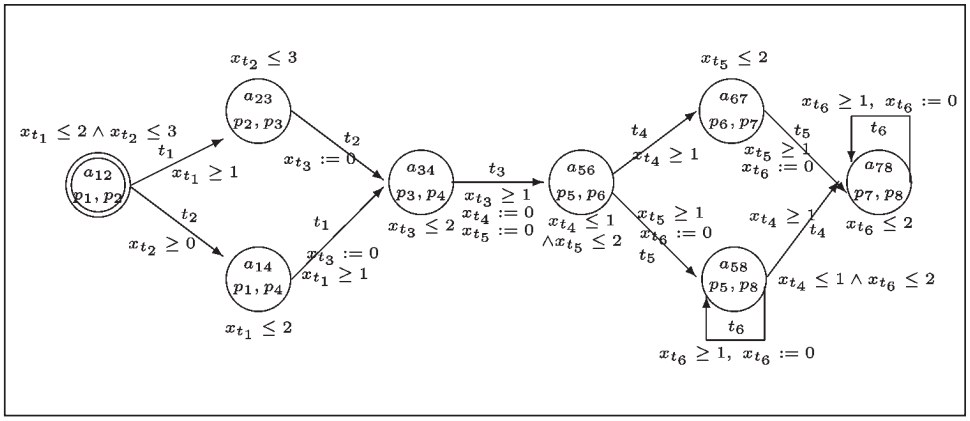

Рис. 8. ВА для ВСП из рис. 7

Пример 8. Временной автомат для ВСП представлен на рис. 8.

Верификация свойств ВСП выполняется путем проверки соответствующих свойств ВА. Если характеризовать ВСП и ВА, то отметим следующее. ВСП определяют класс ВА, слабо бисимуляционно эквивалентных ВСП. Обратное не

имеет места, что следует из результатов работы [20], где показано наличие ВА, для которых не существует ВСП, слабо бисимуляционно эквивалентных с ними. Основная проблема широкого применения этого и подобных ему методов состоит в комбинаторном взрыве числа состояний ВСП и соответствующего ей ВА.

Как для ВА, так и для ВСП существует несколько семантик, описывающих их поведение. Некоторые из этих семантик даны в монографии [34].

ЗАКЛЮЧЕНИЕ

В настоящем обзоре рассмотрены основные применения теории автоматов в различных областях современной науки о вычислениях. Это далеко не полный перечень областей, в которых успешно используются автоматы. Так, не рассматривались задачи идентификации слов [8], приведения общих подвыражений в алгебраических выражениях [35], а также не рассматривались приложения теории автоматов в области унификации, создании аппаратуры вычислительных систем [10, 11], криптографии, лингвистике и других областях. В настоящее время сфера применения конечных автоматов расширяется, и эта тенденция, по-видимому, будет сохраняться в будущем [36].

СПИСОК ЛИТЕРАТУРЫ

1. Глушков В.М. Абстрактная теория автоматов // Успехи мат. наук. — 1961. — № 5. — С. 3–62
2. Глушков В.М., Летичевский А.А., Годлевский А.Б. Методы синтеза дискретных моделей. — Киев: Вища шк., 1983. — 262 с.
3. Кобринский Н.Е., Трахтенброт Б.А. Введение в теорию конечных автоматов. — М.: Физматгиз, 1962. — 286 с.
4. Hopcroft J.E., Montwani J., Ullman J.D. Introduction to autonmata theory and computation (Second addition). — Stanford: Addison Wesley, 2001. — 521 p.
5. Гилл А. Введение в теорию конечных автоматов. — М.: Наука, 1966. — 272 с.
6. Гинзбург С. Математическая теория контекстно-свободных языков. — М.: Мир, 1970. — 326 с.
7. Рабин М.О., Скотт Д. Конечные автоматы и задачи их разрешения // Кибернетический сборник (Старая серия). — М.: Изд-во иностр. лит., 1962. — С. 58–98.
8. Smith B. Computing patterns in strings. — UK: Pearson Education Limited, 2003. — 423 p.
9. Alur R., Dill D.L. A theory of timed automata // TCS. — 1994. — N 126. — P. 183–235.
10. Глушков В.М. Синтез цифровых автоматов. — М.: Физматгиз, 1962. — 562 с.
11. Beyga L., Gajewski T., Miadowicz Z., Siwak P., Stoklosa J., Bergandy J., Mikolajczak B. Algebraic and structural automata theory. — Amsterdam: Elsevier Science Publishers B.V., 1991. — 402 p.
12. Anderson J. Automata theory with modern applications. — Cambridge: Cambridge University Press, 2006. — 255 p.
13. Трахтенброт Б.А., Барздинь Я.М. Конечные автоматы (поведение и синтез). — М.: Наука, 1970. — 400 с.
14. Thomas W. Automata on infinite objects. Handbook on theoretical computer science. — Amsterdam: Elsevier Science Publishers B.V., 1990. — P. 133–191.
15. Gécseg F., Steinby M. Tree automata. — Hungary: Akademiai Kiado, 1984. — 262 p.
16. Gécseg F., Steinby M. Tree languages. Handbook of formal languages. — Amsterdam: Elsevier Science Publishers B.V., 1996. — **3**. — P. 1–68.
17. Comon H., Dauchet M., Gilleron R., Jacquemand F., Lugies D., Tison S., Tommasi M. Tree automata: techniques and applications. TATA, 1999. — 193 p.

18. Фрейвалд Р.В. Распознавание языков на конечных вероятностных многоленточных и многоголовочных автоматах // Проблемы передачи информации. — 1979. — **15**, № 3. — С. 99–106.
19. Бухараев Р.Г. Основы теории вероятностных автоматов. — М.: Наука, 1985. — 394 с.
20. Berard B., Cassez F., Haddad D., Lime D., Roux O.H. Comparison of the expressiveness of time automata and time Petri nets // Proc. of the 3rd Int. Workshop on Formal Analysis and Modeling of Timed Systems (FORMATS'05). — Springer-Verlag. — LNCS. — 2005. — **3829**. — P. 211–225.
21. Тоффоли Т., Марголус Н. Машины клеточных автоматов. — М.: Мир, 1991. — 272 с.
22. Henzinger T.A, Kopke P.W., Puri A., Variaya P. What's decidable about hybrid automata // J. of Computer and System Science. — 1998. — **57**. — P. 94–124.
23. Кривой С.Л. Об алгоритме построения базиса пересечения конечно порожденных свободных групп // Кибернетика. — 1982. — № 4. — С. 5–11.
24. Летичевский А.А., Годлевский А.Б., Кривой С.Л. Об эффективности алгоритма построения базиса подгруппы свободной группы // Кибернетика. — 1981. — № 3. — С. 107–116.
25. Clausen M., Fortenbacher A. Efficient solution of linear diophantine equations // J. Symbolic Computation. — 1989. — **8**, N 1. — P. 201–216.
26. Romeuf J.F. A polinomial algorithm for solvin systems of two linear Diophantine equations // TCS. — 1990. — **74**, N 3. — P. 329–340.
27. Head T. Formal language theory and DNA: an analysis of the generative capacity of specific recombinant behaviors // Bull. Math. Biology. — 1987. — N 49. — P. 737–759.
28. Culik K., Harju T. The regularity of splicing systems and DNA // Proc. ICALP — **89**. — 1989. — N 372. — P. 222–233.
29. Culik K., Harju T. Splicing semigroup of dominoes and DNA // Discrete Applied Mathem. — 1991. — N 31. — P. 261–277.
30. Gatterdam R.W. Splicing systems and regularity // Intern. J. Computer Math. — 1989. — N 31. — P. 63–67.
31. Goode E. Constants and splicing systems. — Binghamton: Dissertation binghamton university, 1999. — 36 p.
32. Ben-Ari M. Mathematical logic for computer science (Second addition). — Springer-Verlag, London-Limited, 2001. — 326 p.
33. Clarke E.M., Grumberg Jr.O., Peled D. Model Checking. The MIT Press: Cambridge, Massachusetts; London, England, 2001. — 356 p.
34. Penczek W., Pо́lrola A. Advanced in verification on time Petri nets and timed automata. Berlin Heidelberg: Springer-Verlag, 2006. — 258 p.
35. Годлевский А.Б., Кривой С.Л. О проектировании эффективных алгоритмов приведения автоматов для некоторых отношений эквивалентности // Кибернетика и системный анализ. — 1989. — № 6. — С. 54–61.
36. Proc. of Conf. "Language and automata theory and applications" // LNCS. — 2009. — **5457**. — 765 p.

Поступила 04.11.2010