УДК 519.178

Б. Ф. Мельников, А. А. Мельникова

МНОГОАСПЕКТНАЯ МИНИМИЗАЦИЯ НЕДЕТЕРМИНИРОВАННЫХ КОНЕЧНЫХ АВТОМАТОВ (ЧАСТЬ I. ВСПОМОГАТЕЛЬНЫЕ ФАКТЫ И АЛГОРИТМЫ)

Аннотация. В первой части настоящей статьи рассматриваются некоторые вспомогательные алгоритмы, необходимые одновременно для двух проблем минимизации недетерминированных конечных автоматов – вершинной и дуговой. Приводится несложный алгоритм минимизации детерминированных автоматов, с помощью которого производится одновременное построение функций разметки состояний. Доказываются вспомогательные утверждения о входных языках состояний базисного автомата, необходимые для алгоритмов эквивалентного преобразования произвольных недетерминированных конечных автоматов.

Ключевые слова: недетерминированный конечный автомат, базисный автомат, алгоритмы эквивалентного преобразования, вершинная минимизация, дуговая минимизация.

Abstract. In the first part of the article, the authors consider some auxiliary algorithms for two problems of minimization of nondeterministic finite automata: vertex-minimization and edge-minimization. The researchers give a simple algorithm for minimization of deterministic automata, which also enables simultaneous construction of state-marking functions. The article proves some auxiliary propositions about input languages of the basis automaton states, necessary for algorithms of equivalent transformation of nondeterministic finite automata.

Key words: nondeterministic finite automaton, basis automaton, algorithms of equivalent transformation, state-minimization, edge-minimization.

Введение

В первой части данной статьи рассматриваются вопросы, связанные с проблемой минимизации недетерминированных конечных автоматов, причем как по числу вершин, так и по числу дуг. Приводятся вспомогательные утверждения и алгоритмы, нужные для решения этих задач, причем необходимые как для точного решения, так и для описания приближенных алгоритмов реального времени (anytime-алгоритмов).

Структура статьи следующая. В разделе 1 приводятся применяемые обозначения – они согласованы с использовавшимися в предыдущих работах авторов ([1–5] и др.). В разделе 2 приводится несложный алгоритм минимизации *детерминированных* автоматов, с его помощью производится *одновременное* построение функций разметки состояний. В разделе 3 доказываются вспомогательные утверждения о входных языках состояний базисного автомата, необходимые для алгоритмов эквивалентного преобразования произвольных недетерминированных конечных автоматов. В заключении приводятся проблемы для дальнейшего решения, а также кратко описывается содержание второй части данной статьи.

Необходимо заранее отметить три следующих важных факта. Во-первых, мы всегда формулируем алгоритмы с целью их возможного использования *в реальных программных системах*; например, мы рассматривали

наши приложения для разработки алгоритмов в [6, 7], где рассматривались anytime-алгоритмы (алгоритмы реального времени) для задачи дуговой минимизации. Во-вторых, мы *не рассматриваем сложность* представленных алгоритмов (минимизационных проблем) также потому, что описанные алгоритмы используются в реальных системах программирования, где более важными являются иные оценки их эффективности [7, 8]. В-третьих, мы иногда используем алгоритм добавления дуги; используя терминологию монографии [8], такое вре́менное ухудшение возможного решения может рассматриваться в качестве примера так называемого порогового локального поиска для задачи дуговой минимизации.

1. Применяемые обозначения

Будем использовать следующие обозначения (они согласованы с [1–5]). Пусть

$$K = (Q, \Sigma, \delta, S, F) - \qquad (1)$$

некоторый конечный автомат (недетерминированный автомат Рабина – Скотта), определяющий регулярный язык $L = \mathcal{L}(K)$. Здесь Q – множество состояний, а $S \subseteq Q$ и $F \subseteq Q$ – множества стартовых и финальных состояний соответственно.

Мы будем рассматривать автомат без ε-переходов, т.е. будем рассматривать функцию переходов δ автомата (1) в виде $\delta: Q \times \Sigma \to \mathcal{P}(Q)$; записью $\mathcal{P}(Q)$ обозначено множество всех подмножеств множества Q.

Через $\mathcal{L}_K^{in}(q)$ и $\mathcal{L}_K^{out}(q)$ обозначим входной и выходной языки состояния q, т.е. языки, определяемые автоматами $(Q, \Sigma, \delta, S, \{q\})$ и $(Q, \Sigma, \delta, \{q\}, F)$ соответственно. А $\mathcal{L}_K^{io}(q)$ – язык, определяемый автоматом $(Q, \Sigma, \delta, \{q\}, \{q\})$.

Для автомата (1) и пары $p, q \in Q$ будем также использовать следующие обозначения. Если $\delta(p, a) \ni r$, то будем писать $p \xrightarrow[\delta]{a} r$ или $p \xrightarrow[K]{a} r$. Будем иногда опускать нижние индексы δ и K, если автомат подразумевается. Аналогичные обозначения будем использовать и для слов. Например, будем писать $p \xrightarrow[K]{u} r$, если существует путь из p в r, помеченный словом $u \in \Sigma^*$; при этом p будет опускаться, когда мы рассматриваем путь из некоторого входа, и т.п.

Зеркальный автомат для заданного автомата (1) будем обозначать K^R, т.е. $K^R = (Q, \Sigma, \delta^R, F, S)$, где $q' \xrightarrow[\delta^R]{a} q''$, если и только если $q'' \xrightarrow[\delta]{a} q'$. Очевидно, что K^R определяет язык L^R.

Для автомата (1) и его состояний $q_1, q_2 \in Q$ будем рассматривать автомат K', чей граф переходов получается из K объединением состояний q_1 и q_2. Автомат K' получается следующим образом (здесь $p, r \in Q$ и $a \in \Sigma$ – любые):

1) полагаем $q_1 \xrightarrow[K']{a} r$, если и только если $q_1 \xrightarrow[K]{a} r$ или $q_2 \xrightarrow[K]{a} r$;

2) аналогично, $p \xrightarrow[K']{a} q_1$, если и только если $p \xrightarrow[K]{a} q_1$ или $p \xrightarrow[K]{a} q_2$;

3) для других пар p, $r \in Q$ полагаем $p \xrightarrow[K']{a} r$, если и только если $p \xrightarrow[K]{a} r$;

4) состояние q_2 и соответствующие ему дуги удаляются.

Кроме того, состояние q_1 является стартовым (финальным) для автомата K', если и только если по крайней мере одно из состояний q_1 и q_2 стартовое (финальное) для автомата K.

Будем обозначать такой автомат через $J^{q_1 q_2}(K)$. Конечно, применение описанной операции J имеет смысл в тех случаях, когда можно доказать, что данный и полученный автоматы (т.е. K и K') эквивалентны.

Для рассматриваемого языка L будем обозначать определяющий его *канонической* конечный автомат записью $\tilde{L}$ [1]. Далее будем считать, что автоматы $\tilde{L}$ и $\tilde{L}^R$ для рассматриваемого языка L следующие:

$$\tilde{L} = (Q_\pi, \Sigma, \delta_\pi, \{s_\pi\}, F_\pi) \text{ и } \tilde{L}^R = (Q_\rho, \Sigma, \delta_\rho, \{s_\rho\}, F_\rho). \quad (2)$$

Рассмотрим некоторые факты из [1–4], используемые в данной статье. Будем использовать две специальные функции (φ^{in} и φ^{out}) – так называемые функции разметки состояний автомата (1), помечающие каждое состояние некоторыми подмножествами состояний $\tilde{L}$ и $\tilde{L}^R$. В данной статье достаточно использовать сокращенную версию их определения: полагаем $\varphi^{in}_K(q) \ni A$ тогда и только тогда, когда существует слово $u \in \Sigma^*$ такое, что $\xrightarrow[K]{u} q$ и $\xrightarrow[BA(L)]{u} A$; аналогично для функции φ^{out}_K. Заметим, что это определение конструктивно, т.е. задает *алгоритм* построения данных функций. Если $\varphi^{in}_K(q) \ni A$ и $\varphi^{out}_K(q) \ni X$, будем писать $q \in \left\langle \begin{smallmatrix} A \\ X \end{smallmatrix} \right\rangle_K$.

Далее мы используем специальное бинарное отношение #, заданное на декартовом произведении $Q_\pi \times Q_\rho$ (см. [2] и др.); отметим, что это отношение определяется заданным языком, а не конкретным автоматом для его задания. Проще всего указанное отношение может быть получено при построении функции φ^{in}, рассматриваемой для автомата, зеркального к каноническому для данного языка. Далее будет приведен подробный пример работы таких алгоритмов.

Отметим, что пары состояний отношения # могут рассматриваться как состояния базисного автомата для данного языка ([2, 4] и др.); приведем его подробное определение. Для заданного регулярного языка L определим автомат $BA(L) = (T, \Sigma, \delta_T, s_T, F_T)$ следующим образом:

• T – множество пар вида $\begin{smallmatrix} A \\ X \end{smallmatrix}$ таких, что $A \in Q_\pi$, $X \in Q_\rho$ и $A \# X$. При этом мы будем писать $A = \alpha\left(\begin{smallmatrix} A \\ X \end{smallmatrix}\right)$ и $X = \beta\left(\begin{smallmatrix} A \\ X \end{smallmatrix}\right)$.

[1] Аналогично [1–4] мы считаем каноническим автоматом детерминированный автомат, содержащий минимально возможное число состояний. При этом также аналогично [1–4] мы *не* требуем всюду определенности этого автомата и, следовательно, *не* включаем в него (единственное) возможное «дохлое состояние» (“dead state”).

• Функцию δ_T определим следующим образом. Для всех состояний $T_1, T_2 \in T$ и буквы $a \in \Sigma$ полагаем $T_2 \in \delta_T(T_1, a)$ тогда и только тогда, когда $\delta_\pi\left(\alpha(T_1), a\right) = \{\alpha(T_2)\}$ и $\delta_\rho\left(\beta(T_2), a\right) = \{\beta(T_1)\}$.

• Считаем, что $T \in S_T$ тогда и только тогда, когда $\alpha(T) = s_\pi$; аналогично: $T \in F_T$ тогда и только тогда, когда $\beta(T) = s_\rho$.

Базисный конечный автомат [2] является инвариантом регулярного языка. При этом во многих теоретических и практических задачах теории формальных языков рассматривать базисные автоматы существенно удобнее, чем канонические.

2. Подробный пример – построение функций разметки с помощью канонического автомата

На русском языке подробное рассмотрение примера авторами не публиковалось (в частности, его не было в монографии [9]). Менее подробно построение функций разметки в процессе построения канонического автомата рассматривалось в нескольких статьях на английском языке, однако здесь мы исправили некоторые неточности и привели наиболее подробные варианты алгоритмов.

Фактически приведенный здесь алгоритм является специальной модификацией алгоритма Хопкрофта [10], он позволяет строить не только эквивалентный канонический автомат, но и – одновременно – некоторые вспомогательные объекты. Однако, как было отмечено выше, вопросы сложности алгоритмов в данной статье не рассматриваются.

Итак, в предыдущих статьях [1–5] алгоритмы построения бинарного отношения # и базисного автомата не рассматривались подробно. Отметим, что для этих алгоритмов даже процедуру построения канонического автомата желательно делать несколько иной – отличающейся от «классической». В связи с этим мы рассмотрим такие алгоритмы детально. Для заданного конечного автомата мы рассмотрим[1] приемлемый алгоритм *одновременного* построения:

- эквивалентного ему канонического автомата;
- функции разметки φ^{in2};
- бинарного отношения #;
- множеств блоков (гридов).

Таким образом, мы детально рассмотрим варианты применения классических алгоритмов теории конечных автоматов к построению «наших» объектов для заданного регулярного языка[3].

Мы будем рассматривать очень интересный (для обеих вышеупомянутых минимизационных проблем, вершинной и дуговой) пример конечного автомата. В частности, автомат, заданный на рис. 1, имеет следующие свойства:

---

[1] Частично – в данном разделе, а частично – во второй части данной статьи.

[2] А также φ^{out}. Однако *полного* алгоритма построения последней функции мы приводить не будем.

[3] Подробное описание алгоритма построения базисного автомата можно найти, например, в [4].

– и число его состояний, и число его дуг *меньше* числа состояний и числа дуг эквивалентного канонического автомата;

– то же самое верно и для зеркального автомата.

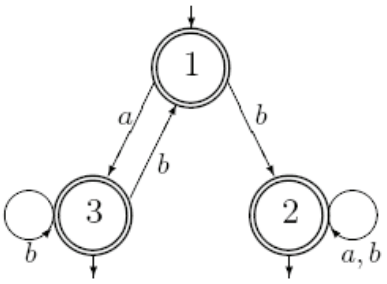

Рис. 1

Более того, этот автомат – минимальный среди всех конечных автоматов с тремя состояниями, обладающими всеми указанными свойствами[1]. Заметим также, что «связанные» примеры из [12] используют алфавит, содержащий 2^N букв, где N – число состояний соответствующего канонического автомата[2].

Для проведения процедуры канонизации запишем данный автомат в виде табл. 1. А рассматривая содержимое клеток табл. 1 как *состояния* (а *не множества* состояний), мы получаем табл. 2.

Таблица 1

	K	a	b
$\rightarrow$	1	3	2
$\leftarrow$	2	2	2
$\leftarrow$	3	–	1, 3

Таблица 2

		a	b
$\rightarrow$	1	3	2
$\leftarrow$	3	–	1, 3
$\leftarrow$	2	2	2
$\leftarrow$	1, 3	3	1, 2, 3
$\leftarrow$	1, 2, 3	2, 3	1, 2, 3
$\leftarrow$	2, 3	2	1, 2, 3

[1] При доказательстве мы использовали компьютерную программу, рассматривающую все автоматы с тремя состояниями. Возможно и «непереборное» доказательство на основе результатов статьи [11].

[2] Конечно, эти примеры связаны с несколько иными предметными областями. Например, авторам вообще не известны монографии, где бы ставилась проблема дуговой минимизации.

Обозначив данные множества состояний как состояния *нового* автомата, мы получаем эквивалентный детерминированный автомат (табл. 3).

Таблица 3

		a	b
$\rightarrow$	A	B	C
$\leftarrow$	B	–	D
$\leftarrow$	C	C	C
$\leftarrow$	D	B	E
$\leftarrow$	E	F	E
$\leftarrow$	F	C	E

Теперь получим множества эквивалентных состояний. Для этого рассмотрим бинарное отношение $\succ$, определенное на множестве, включающем, во-первых, все пары состояний $\tilde{L}$; во-вторых, специальный символ $\mathcal{F}$. Этот символ обозначает, что любая пара состояний, находящаяся с ним в отношении $\succ$, состоит из заведомо *неэквивалентных* состояний, причем этот факт известен с самого начала.

Для двух различных состояний G и H полагаем $\{G, H\} \succ \mathcal{F}$ (мы будем ниже писать просто GH вместо $\{G, H\}$) тогда и только тогда, когда финальным является в точности одно из этих состояний. Для четырех состояний G_1, H_1, G_2 и H_2 (где $G_1 \neq H_1$ и $G_2 \neq H_2$) мы полагаем $G_1H_1 \succ G_2H_2$ (или $G_1H_1 \succ H_2G_2$, что то же самое), если существует по крайней мере одна буква c такая, что $G_1 \xrightarrow{c} G_2$ и $H_1 \xrightarrow{c} H_2$.

В данном примере для описания бинарного отношения $\succ$ мы должны были *бы* использовать таблицу, имеющую 20 строк (для пар AB, …, EF) и 21 столбец (для тех же пар и символа $\mathcal{F}$). Но такая таблица слишком велика, поэтому мы будем использовать ее упрощение, используя, в частности, тот факт, что ни A, ни B не могут быть эквивалентны никаким другим состояниям:

- A – потому что оно финальное (в отличие от любого другого состояния);
- B – потому что оно не имеет выходящих из него a-дуг (также в отличие от других состояний), а бесполезных состояний в автомате нет.

Итак, рассмотрим таблицу с шестью строками (для пар CD,…,EF) и семью столбцами (табл. 4). В ее клетках вместо 1 и 0 мы записываем любой из двух подходящих переходов последнего автомата (этот факт символизирует, что отношение $\succ$ истинно) или «–» (символизирующий отсутствие таких переходов, при этом отношение $\succ$ ложно). Аналогично для последнего столбца: все состояния в этой таблице финальные, поэтому мы записываем возможный переход в A или B (которые не эквивалентны другим состояниям) для ровно одного из двух состояний рассматриваемой строки.

Запишем в пустых клетках 0, а в остальных 1; после этого, выполнив транзитивное замыкание $\succ^+$, получаем табл. 5. Согласно ее последнему

столбцу[1], состояния C, E и F эквивалентны, а состояние D им не эквивалентно.

Таблица 4

$\succ$	CD	CE	CF	DE	DF	EF	$\mathcal{F}$
CD	–	$C \xrightarrow{b} C$ $D \xrightarrow{b} E$	–	–	–	–	$C \xrightarrow{a} C$ $D \xrightarrow{a} B$
CE	–	–	$C \xrightarrow{a} C$ $E \xrightarrow{a} F$	–	–	–	–
CF	–	$C \xrightarrow{b} C$ $F \xrightarrow{b} E$	–	–	–	–	–
DE	–	–	–	–	–	–	$D \xrightarrow{a} B$ $E \xrightarrow{a} F$
DF	–	–	–	–	–	–	$D \xrightarrow{a} B$ $F \xrightarrow{a} C$
EF	–	–	$E \xrightarrow{a} F$ $F \xrightarrow{a} C$	–	–	–	–

Таблица 5

$\succ^+$	CD	CE	CF	DE	DF	EF	$\mathcal{F}$
CD	0	1	1	0	0	0	1
CE	0	1	1	0	0	0	0
CF	0	1	1	0	0	0	0
DE	0	0	0	0	0	0	1
DF	0	0	0	0	0	0	1
EF	0	1	1	0	0	0	0

Объединяя состояния, мы из табл. 3 получим табл. 6 эквивалентного канонического автомата. В ней отмечены состояния исходного автомата в соответствии с табл. 2. Однако обычно необходимо и обратное соответствие, согласно которому мы получаем следующую функцию φ^{in}:

$$\varphi_K^{in}(1) = \{A, C, D\},\ \varphi_K^{in}(2) = \{C\},\ \varphi_K^{in}(3) = \{B, C, D\}. \qquad (3)$$

Таблица 6

	$\widetilde{L}$		a	b
$\rightarrow$	A	1	B	C
$\leftarrow$	B	3	–	D
$\leftarrow$	C	1, 2, 3	C	C
$\leftarrow$	D	1, 3	B	C

[1] Отметим, что мы получаем несколько единиц на главной диагонали, но это не имеет значения.

Итак, на рис. 2 и 3 показаны автомат $\tilde{L}$ и зеркальный к нему $\left(\tilde{L}\right)^R$.

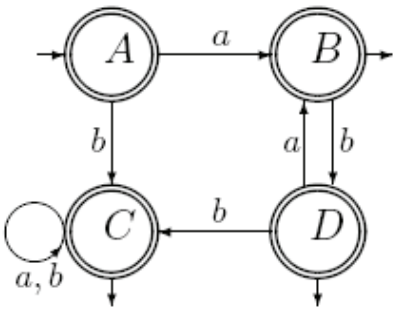

Рис. 2

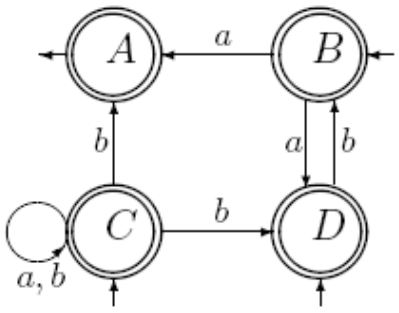

Рис. 3

Аналогично сделанному ранее запишем автомат в виде табл. 7.

Таблица 7

	$(\tilde{L})^R$	a	b
$\leftarrow$	A	–	–
$\rightarrow$	B	A, D	–
$\rightarrow$	C	C	A, C, D
$\rightarrow$	D	–	B

Рассматривая содержание клеток табл. 7 *как состояния* (а *не* множества состояний), получаем табл. 8. Переобозначив в ней множества состояний как состояния нового автомата, мы получаем эквивалентный детерминированный автомат (табл. 9)

Таблица 8

		a	b
$\rightarrow$	B, C, D	A, C, D	A, B, C, D
$\leftarrow$	A, C, D	C	A, B, C, D
$\leftarrow$	A, B, C, D	A, C, D	A, B, C, D
	C	C	A, C, D

Рассмотрение этого примера (а именно построение бинарного отношения # и множества блоков рассматриваемого автомата) будет продолжено во второй части данной статьи.

3. О входных языках состояний базисного автомата

В этом разделе доказываются вспомогательные утверждения о входных языках состояний базисного автомата, необходимые для алгоритмов эквивалентного преобразования произвольных недетерминированных конечных автоматов.

Таблица 9

		$\widetilde{L^R}$	a	b
$\rightarrow$		X	Y	Z
	$\leftarrow$	Y	U	Z
	$\leftarrow$	Z	Y	Z
		U	U	Y

Утверждение 1. Для любого языка L и для любых состояний $A \in Q_\pi$ и $X \in Q_\rho$ (таких, что $A\#X$) выполнено следующее:

$$\mathcal{L}^{in}_{BA(L)}\binom{A}{X} = \mathcal{L}^{in}_{\tilde{L}}(A) \text{ и } \mathcal{L}^{out}_{BA(L)}\binom{A}{X} = (\mathcal{L}^{in}_{\tilde{L}^R}(X))^R. \quad (4)$$

Доказательство. Докажем только первое равенство, второе получается аналогично. Условие $\mathcal{L}^{in}_{BA(L)}\binom{A}{X} \subseteq \mathcal{L}^{in}_{\tilde{L}}(A)$ непосредственно следует из определения автомата $BA(L)$. Поэтому достаточно доказать, что если для некоторого слова w верно $w \in \mathcal{L}^{in}_{\tilde{L}}(A)$, то и $w \in \mathcal{L}^{in}_{BA(L)}\binom{A}{X}$; докажем этот факт индукцией по $|w|$.

База индукции, $|w| = 0$, т.е. $w = \varepsilon$. В этом случае $A = s_\pi$, поэтому (по определению базисного автомата) $\binom{A}{X} \in S_T$, т.е. $w = \varepsilon \in \mathcal{L}^{in}_{BA(L)}\binom{A}{X}$.

Шаг индукции. Пусть $w = w'a$, где $w = w'a$, $w' \in \Sigma^*$, $a \in \Sigma$, и пусть $w \in \mathcal{L}^{in}_{\tilde{L}}(A)$. Поскольку $A\#X$, существуют слова $u, v \in \Sigma^*$, такие что $uv \in L$, и при этом $u \in \mathcal{L}^{in}_{\tilde{L}}(A)$, $v^R \in \mathcal{L}^{in}_{\tilde{L}^R}(X)$. Поскольку и для слова w выполнено условие $w \in \mathcal{L}^{in}_{\tilde{L}}(A)$, мы получаем, что $wv \in L$, т.е. $w'av \in L$.

Пусть состояние $B \in Q_\pi$ такое, что $w' \in \mathcal{L}^{in}_{\tilde{L}}(B)$, а $Y \in Q_\rho$ – такое, что $v^R a \in \mathcal{L}^{in}_{\tilde{L}^R}(Y)$; оба эти состояния ($B$ и Y) существуют, поскольку $w'av \in L$. Поэтому $B\#Y$, и согласно предположению индукции $w' \in \mathcal{L}^{in}_{BA(L)}\binom{B}{Y}$. А вследствие $\binom{B}{Y} \xrightarrow[BA(L)]{a} \binom{A}{X}$ мы получаем, что $w \in \mathcal{L}^{in}_{BA(L)}\binom{A}{X}$. □

Утверждение 2. Пусть для некоторого регулярного языка L и определяющего его автомата (1) выполнены следующие условия: $q \in Q$, $A \in \varphi^{in}_K(Q)$, $X \in \varphi^{out}_K(Q)$, $v \in \mathcal{L}^{in}_K(q)$, $u \in \mathcal{L}^{in}_{BA(L)}\binom{A}{X}$ и $w \in \mathcal{L}^{out}_{BA(L)}\binom{A}{X}$. Тогда для любого $i \geq 0$ выполнено следующее: $uv^i w \in L$.[1]

Доказательство. Поскольку $A \in \varphi^{in}_K(Q)$ и $X \in \varphi^{out}_K(Q)$, то существуют слова x и y такие, что $x \in \mathcal{L}^{in}_{\tilde{L}}(A) \cap \mathcal{L}^{in}_K(q)$, $y \in \mathcal{L}^{in}_{\tilde{L}^R}(X) \cap \mathcal{L}^{in}_{K^R}(q)$. Поскольку

[1] Важно отметить, что, вообще говоря, $u \notin \mathcal{L}^{in}_K(q)$ и $w \notin \mathcal{L}^{out}_K(q)$.

$v \in \mathcal{L}_K^{io}(q)$, слово xv^iy^R принимается автоматом K (т.е. $L \ni xv^iy^R$), а отсюда следует, что $y(v^i)^R x^R \in L^R$. Согласно утверждению 1 $w^R \in \mathcal{L}_{\tilde{L}^R}^{in}(X)$, поэтому $w^R(v^i)^R x^R \in L^R$. Это означает, что $xv^iw \in L$. Также согласно утверждению 1 $u \in \mathcal{L}_{\tilde{L}}^{in}(A)$, поэтому $uv^iw \in L$. □

Заключение

Опишем некоторые возможные проблемы для будущего решения:

1) подробное исследование сложности приведенного нами выше варианта алгоритма Хопкрофта. Фактически надо показать, что в случае *отдельных алгоритмов* для канонизации автомата и для построения вспомогательных объектов (вместо алгоритмов, рассматривавшихся в разделе 3) общая сложность вычислений возрастет;

2) формулировка достаточных условий возможного *удаления* дуги из рассматриваемого автомата с сохранением описываемого языка. Здесь имеются в виду условия, *не требующие* построения эквивалентных канонических автоматов (для заданного языка и полученного автомата);

3) доказательство того, что подобный процесс удаления дуг *не может* являться альтернативой алгоритму дуговой минимизации;

4) наоборот – формулирование условий для *невозможности* удаления конкретной дуги из заданного автомата[1].

Авторы надеются, что возможное решение этих подзадач поможет решению различных проблем теории формальных языков и конечных автоматов.

Во второй части данной статьи будет продолжено подробное рассмотрение примера, начатого в первой части: в процессе выполнения процедуры канонизации автомата будут получены бинарное отношение # и множество блоков. Будут приведены два алгоритма объединения состояний недетерминированного автомата. На основе этих алгоритмов будут сформулированы сокращенный вариант алгоритма дуговой минимизации, а также алгоритм добавления дуги.

Список литературы

1. **Melnikov, B.** Edge-minimization of non-deterministic finite automata / B. Melnikov, A. Melnikova // The Korean J. Comp. and Appl. Math. – 2001. – V. 8, № 3. – P. 469–479.
2. **Melnikov, B.** Some properties of the basis finite automaton / B. Melnikov, A. Melnikova // The Korean J. Comp. and Appl. Math. – 2002. – V. 9, № 1. – P. 131–150.
3. **Melnikov, B.** Possible edges of a finite automaton defining a given regular language / B. Melnikov, N. Sciarini-Guryanova // The Korean J. Comp. and Appl. Math. – 2002. – V. 9, № 2. – P. 475–485.

[1] Нельзя утверждать, что множество дуг, которое может быть удалено, является дополнением множества дуг, которые удалять нельзя: в обеих этих подзадачах мы ищем *достаточные* условия. Отметим еще, что мы часто рассматриваем не конечный автомат, а специальную «автоматоподобную» структуру, подробнее см. [3]. И именно для таких структур нужны обе эти задачи.

4. **Melnikov, B.** A new algorithm of constructing the basis finite automaton / B. Melnikov, A. Melnikova // Informatica (Lithuanian Acad. Sci. Ed.). – 2002. – V. 13, № 3. – P. 299–310.
5. **Мельников, Б.** Однозначные конечные автоматы / Б. Мельников // Известия Высших учебных заведений. Поволжский регион. Технические науки. – 2002. – № 2. – С. 9–14.
6. **Melnikov, B.** Discrete optimization problems – some new heuristic approaches / B. Melnikov // The 8th Int. Conf. on High-Perfomance Computing and Grid in Asia-Pacific Region, IEEE Comp. Soc. Press Ed. – 2005. – P. 73–80.
7. **Мельников, Б.** Мультиэвристический подход к задачам дискретной оптимизации / Б. Мельников // Кибернетика и системный анализ (НАН Украины). – 2006. – № 3. – P. 32–42.
8. **Hromkovic, J.** Algorithmics for Hard Problems / J. Hromkovic // Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics. – Springer, 2004. – 548 p.
9. **Мельников, Б.** Недетерминированные конечные автоматы / Б. Мельников. – Тольятти : Изд-во ТГУ, 2009. – 160 с.
10. **Hopcroft, J.** An $n \cdot \log n$ algorithm for minimizing the states in a finite automaton / J. Hopcroft // The Theory of Machines and Computations / Z. Kohavi (ed.). – Academic Press, 1971. – P. 189–196.
11. **Мельников, Б.** Алгоритмы эквивалентного преобразования недетерминированных конечных автоматов / Б. Мельников, М. Сайфуллина // Известия высших учебных заведений. Математика. – 2009. – № 4. – С. 67–71.
12. **Брауэр, В.** Введение в теорию конечных автоматов / В. Брауэр. – М. : Радио и связь, 1987. – 392 с.

Мельников Борис Феликсович
доктор физико-математических наук, профессор, кафедра прикладной математики и информатики, Тольяттинский государственный университет

E-mail: B.Melnikov@tltsu.ru

Melnikov Boris Felixovich
Doctor of physical and mathematical sciences, professor, sub-department of applied mathematics and informatics, Tolyatti State University

Мельникова Александра Александровна
старший преподаватель, кафедра высшей математики, Димитровградский филиал Ульяновского государственного университета

E-mail: avahi@mail.ru

Melnikova Alexandra Alexandrovna
Senior lecturer, sub-department of higher mathematics, Dimitrovgrad branch of Ulyanovsk State University

---

УДК 519.178

Мельников, Б. Ф.

Многоаспектная минимизация недетерминированных конечных автоматов (Часть I. Вспомогательные факты и алгоритмы) / Б. Ф. Мельников, А. А. Мельникова // Известия высших учебных заведений. Поволжский регион. Физико-математические науки. – 2011. – № 4 (20). – С. 59–69.