О дугах конечных автоматов для описания алгоритмов построения Ватерлоо-подобных автоматов[1]

В. Н. Долгов, аспирант
Тольяттинский государственный университет
terenga74@mail.ru
Н. В. Софонова, аспирант
Тольяттинский государственный университет
natashamilkova@yandex.ru

Алгоритмы минимизации недетерминированных конечных автоматов, как известно, усложняет существование для некоторых автоматов конструкций как из примера найденного Камедой и Вайнером и названного Ватерлоо. До применения практических алгоритмов вершинной минимизации недетерминированных конечных автоматов на основе автомата Ватерлоо, а также т.н. полного автомата для соответствующего языка, желательно строить подобную конструкцию автоматически. В настоящей статье рассматривается необходимое для этого построения сравнение дуг полного автомата с дугами универсального автомата и с дугами базисного автомата.

Ключевые слова: недетерминированные конечные автоматы, алгоритмы минимизации, универсальный автомат, базисный автомат, полный автомат.

1. Введение

Как известно, для регулярного языка имеются разные инварианты, например, канонический, базисный и универсальный конечные автоматы – [1,2,3,4] и др. В работе [2] предлагаются алгоритмы построения т. н. автомата $\mathcal{COM}(L)$ и приводится доказательство того, что этот автомат совпадает, с точностью до переобозначения состояний, с универсальным автоматом. Фактически в той статье приводятся несколько различных алгоритмов построения универсального автомата, один из них используется в настоящей статье.

Алгоритмы минимизации недетерминированных конечных автоматов (определения решаемых ими задачи можно найти в [5, 6,

[1] © В. Н. Долгов, Н. В. Софонова, 2015

7, 8, 9] и мн. др.), как известно, усложняет существование для некоторых недетерминированных конечных автоматов случаев, когда покрывающее множество блоков [2] определяет автомат (покрывающий автомат), *не* эквивалентным исходному. Пример такого случая можно найти в статье [10] Камеды и Вайнера. Он носит название Ватерлоо. В нашей терминологии подробное описание этого примера изложено в [11], подобную конструкцию мы называем *walibad* (от выражения "Waterloo-like badness"), а любое покрывающее подмножество множества блоков, включающее *не все* блоки, называем *npomo-walibad*. Алгоритмы минимизации недетерминированных конечных автоматов использующие свойства покрывающих подмножеств при работе с walibad'ом могут выдать минимизированный автомат не эквивалентный исходному. Поэтому хотелось бы заранее (до применения алгоритмов минимизации) знать ответ на вопрос – является ли автомат, который мы хотим минимизировать, walibad'ом.

В статье рассматривается полный автомат $L^{\#}$, который необходим для автоматического построения новых конструкций walibad на основе некоторого примера walibad'а и, следовательно, соответствующей ему таблицы бинарного отношения # (см. [2, 3, 6] и др.). Полный автомат определяется согласно такой таблицы бинарного отношения # в [3]. Описываются помечающие функции из [11] заданные для множества дуг полного автомата $L^{\#}$ необходимые для построения walibad'ов.

Описывается построение автомата $\mathcal{COM}(L)$ для заданного автомата K, на основе возможных дуг которого строится покрывающий автомат. Этот покрывающий автомат может быть не эквивалентным исходному автомату K, т.е. привести нас к ситуации когда мы нашли walibad. Доказывается что, дуги автомата $\mathcal{COM}(L)$ соответствуют дугам полного автомата $L^{\#}$.

Описывается базисный автомат и функция $c_{\#}$, помечающая во множестве дуг полного автомата дуги базисного автомата. Показывается что, объединение букв полного автомата на основе помечающей функции $c_{\#}$ даёт нам walibad. Наша цель построить помечающую функцию $c_{\#}$ переборным алгоритмом. Для этого доказы-

[2] Здесь и далее связанная с блоками терминология даётся согласно [6]. При этом покрывающие автоматы строятся согласно [2, разд. 4].

вается что, для каждой дуги базисного автомата $\mathcal{BA}(L)$ существует соответствующая дуга полного автомата $L^{\#}$.

2. Предварительные свéдения

Мы будем использовать обозначения и некоторые факты из [1,2, 3,12,13,14]. Будем рассматривать недетерминированный конечный автомат

$$K = (\, Q, \Sigma, \delta, S, F \,) \tag{1}$$

без ε-переходов, т.е., мы будем рассматривать функцию переходов δ автомата (1) как

$$\delta : Q \times \Sigma \to \mathcal{P}(Q)\,.$$

Будем считать регулярный язык L заданным и для него использовать обозначения, определённые в процитированных статьях; например, канонические автоматы для языков L и L^R (т.е., $\widetilde{L}$ и $\widetilde{L^R}$) будут следующими:

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

(Мы не будем рассматривать язык $L = \varnothing$, поэтому оба этих автомата *имеют* начальные состояния.)

Функция разметки $\varphi_K^{in} : Q \to \mathcal{P}(Q_\pi)$ определяется на множествах состояний автоматов $\widetilde{L}$ и $\widetilde{L^R}$ таким образом:

$$\varphi_K^{in}(q) \ni \widetilde{q} \quad \text{если и только если} \quad \mathcal{L}_K^{in}(q) \cap \mathcal{L}_{\widetilde{L}}^{in}(\widetilde{q}) \neq \varnothing\,.$$

Функция разметки

$$\varphi_K^{out} : Q \to \mathcal{P}(Q_\rho)$$

определяется аналогично для автоматов K^R и $\widetilde{L^R}$.

Бинарное отношение $\# \subseteq Q_\pi \times Q_\rho$ определяется для пар состояний автоматов $\widetilde{L}$ и $\widetilde{L^R}$ следующим образом:

$$A \# X \;\; \text{если и только если} \;\; \Big(\exists uv \in L\Big) \Big(u \in \mathcal{L}_{\widetilde{L}}^{in}(A)\,,\; v^R \in \mathcal{L}_{\widetilde{L^R}}^{in}(X)\Big)\,.$$

3. Полный автомат

Рассмотрим *полный автомат* $L^{\#}$ (см. [3]):

$$L^{\#s_\pi s_\rho} = L^{\#} = (Q_\pi, \Sigma_\#, \delta_\#, \{s_\pi\}, F_\pi), \quad \text{где}$$

- $F_\pi = \left\{ f_\pi \in Q_\pi \,\middle|\, f_\pi \# s_\rho \right\}$ – множество финальных состояний;
- $\Sigma_\# = \left\{ a_{\ {}_X^A} \,\middle|\, A \in Q_\pi, X \in Q_\rho \right\}$ – алфавит;
- $\delta_\#(A_1, a_{\ {}_X^{A_2}}) = \begin{cases} \{A_2\}, & \text{если } A_1 \# X \\ \varnothing, & \text{иначе} \end{cases}$ – функция переходов.

Приведем таблицу бинарного отношения # для автомата Ватерлоо.

	X	Y	Z	U	V	W	P	Q	R	S
A		#						#		
B		#			#					
C				#	#					
D						#	#			
E	#					#	#			
F	#		#				#		#	
G			#						#	#
H			#							#

Построенная по ней таблица переходов автомата $L^{\#}$ для языка

Ватерлоо включает 80 столбцов, которые разбиты на 5 таблиц:

	$\overset{A}{X}$	$\overset{B}{X}$	$\overset{C}{X}$	$\overset{D}{X}$	$\overset{E}{X}$	$\overset{F}{X}$	$\overset{G}{X}$	$\overset{H}{X}$	$\overset{A}{Y}$	$\overset{B}{Y}$	$\overset{C}{Y}$	$\overset{D}{Y}$	$\overset{E}{Y}$	$\overset{F}{Y}$	$\overset{G}{Y}$	$\overset{H}{Y}$
$\to A$									A	B	C	D	E	F	G	H
B									A	B	C	D	E	F	G	H
C																
D																
$\leftarrow E$	A	B	C	D	E	F	G	H								
$\leftarrow F$	A	B	C	D	E	F	G	H								
G																
H																

	$\overset{A}{Z}$	$\overset{B}{Z}$	$\overset{C}{Z}$	$\overset{D}{Z}$	$\overset{E}{Z}$	$\overset{F}{Z}$	$\overset{G}{Z}$	$\overset{H}{Z}$	$\overset{A}{U}$	$\overset{B}{U}$	$\overset{C}{U}$	$\overset{D}{U}$	$\overset{E}{U}$	$\overset{F}{U}$	$\overset{G}{U}$	$\overset{H}{U}$
$\to A$																
B																
C									A	B	C	D	E	F	G	H
D																
$\leftarrow E$																
$\leftarrow F$	A	B	C	D	E	F	G	H								
G	A	B	C	D	E	F	G	H								
H	A	B	C	D	E	F	G	H								

	A V	B V	C V	D V	E V	F V	G V	H V	A W	B W	C W	D W	E W	F W	G W	H W
$\to A$																
B	A	B	C	D	E	F	G	H								
C	A	B	C	D	E	F	G	H								
D									A	B	C	D	E	F	G	H
$\leftarrow E$									A	B	C	D	E	F	G	H
$\leftarrow F$																
G																
H																

	A P	B P	C P	D P	E P	F P	G P	H P	A Q	B Q	C Q	D Q	E Q	F Q	G Q	H Q
$\to A$									A	B	C	D	E	F	G	H
B																
C																
D	A	B	C	D	E	F	G	H								
$\leftarrow E$	A	B	C	D	E	F	G	H								
$\leftarrow F$	A	B	C	D	E	F	G	H								
G																
H																

	A R	B R	C R	D R	E R	F R	G R	H R	A S	B S	C S	D S	E S	F S	G S	H S
$\to A$																
B																
C																
D																
$\leftarrow E$																
$\leftarrow F$	A	B	C	D	E	F	G	H								
G	A	B	C	D	E	F	G	H	A	B	C	D	E	F	G	H
H									A	B	C	D	E	F	G	H

Для выделения из полного автомата walibad'a нам необходимо по какому-то правилу объединить некоторые буквы полного автомата. Эти правила из статьи [11] назовем раскраской и поясним их применение для получения walibad'a.

О п р е д е л е н и е 1. *Пусть Σ' – некоторый алфавит. Рассмотрим функцию*

$$c : \Sigma' \times \Sigma_{\#} \to \mathcal{P}(Q_{\pi}) \,.$$

Назовчм с ***раскраской переходов*** *автомата $L^{\#}$, если для любых $a \in \Sigma'$, $a_{X_1 A_2} \in \Sigma_{\#}$ и $A_1 \in Q_{\pi}$ выполняется следующее условие:*

$$A_1 \in c(a, a_{X_1 A_2}) \Rightarrow A_1 \# X_1 \quad (\textit{т.е. } \delta_{\#}(A_1, a_{X_1 A_2}) \neq \varnothing).$$

Наглядный смысл функции раскраски таков: условие

$$c(a, a_{X_1 A_2}) \neq \varnothing$$

означает, что мы пометили буквой a столбец функции переходов $\delta_{\#}$, соответствующий букве $a_{X_1 A_2}$, и, кроме того, отметили в нчм некоторые переходы, соответствующие множеству $c(a, a_{X_1 A_2})$. При этом значение функции переходов $\delta_{\#}(A_1, a_{X_1 A_2}) = \{A_2\}$ подразумевается.

О п р е д е л е н и е 2. *Назовём с* ***допустимой раскраской****, если выполняется следующее условие:*

$$A_1 \in c(a, a_{X_2 A_3}) \quad \text{и} \quad A_1 \in c(a, a_{X_4 A_5}) \quad \text{влечёт} \quad A_3 = A_5.$$

Суть допустимой раскраски состоит в том, что все помеченные переходы (значения функции переходов $\delta_{\#}$) для вершины A и буквы a совпадают.

4. Соответствие дуг автомата $\mathcal{COM}(L)$ дугам полного автомата

Пусть задан некоторый регулярный язык L. Рассмотрим автомат $\mathcal{COM}(L)$; его определение и некоторые связанные с ним понятия можно найти в [2]. Приведём два условия существования перехода $\mathcal{B}_1 \xrightarrow[\delta_Q]{a} \mathcal{B}_2$:

$$\big(\forall p \in \alpha(\mathcal{B}_1)\big)\ \big(\delta_\pi(p, a) \in \alpha(\mathcal{B}_2)\big) \tag{2}$$

$$\big(\forall r \in \beta(\mathcal{B}_2)\big)\ \big(\delta_\rho(r, a) \in \beta(\mathcal{B}_1)\big), \tag{3}$$

Рассмотрим один из алгоритмов построения автомата $\mathcal{COM}(L)$ из [2] в нашей терминологии.

Для получения одного из вариантов автомата $\mathcal{COM}(L)$ сначала проводим процедуры детерминизации и канонизации исходного автомата K (1). Далее для получения таблицы # применяем процедуры детерминизации, канонизации и переобозначения состояний к

$(\widetilde{K})^R$. Состояния из множества переобозначенных состояний находятся в отношении $\#$ с состоянием, в которое это множество переобозначили. Далее используем получившиеся из таблицы $\#$ блоки как состояния автомата $\mathcal{COM}(L)$.

Для q – состояния-блока, $\alpha(q)$ – соответствующее множество состояний $\widetilde{L}$ и $\beta(q)$ – соответствующее множество состояний $\widetilde{L^R}$. Начнём строить дугу из состояния-блока во множество блоков удовлетворяющую условиям существования дуги. Для любого состояния соответствующего состоянию $\widetilde{L}$ из множества по компоненте α первого (выходного) блока $\mathcal{B}_1$ переход по рассматриваемой букве в $\widetilde{L}$ приводит в множество состояний, которое мы обозначим $\delta(Q)$ для каждого состояния-блока. А все блоки, в которые входит это множество $\delta(Q)$ состояний $\widetilde{L}$ по компоненте α, записываем в блоки-кандидаты по α. Т. е. эти блоки-кандидаты могут быть вторыми (входными) блоками $\mathcal{B}_2$ по компоненте α, что необходимо проверить для каждого кандидата по компоненте β.

По таблице для $\widetilde{L^R}$, которую мы строим детерминируя, канонизируя и переобозначая $(\widetilde{K})^R$ проверяем второе из условий (3) существования дуги автомата $\mathcal{COM}(L)$. Т.е. для каждого второго (в $\widetilde{L^R}$, во втором условии существования дуги – выходного по компоненте β) блока $\mathcal{B}_2$ (пока являющегося только кандидатом) смотрим куда ведет переход в $\widetilde{L^R}$ из состояний из множества по компоненте β по рассматриваемой букве. Блок-кандидат становится блоком-переходом, если мы попадаем по компоненте β в первый (выходной в $\widetilde{L^R}$) блок $\mathcal{B}_1$.

Определение 3. *Множество букв $a_{X_1 A_2}$, где $A_2 \in Q_\pi$, по которым есть переход из A_1 назовём* ***множеством букв соответствующих элементу*** *$A_1\#X_1$ бинарного отношения $\#$.*

Теорема 1. *Для каждой дуги автомата $\mathcal{COM}(L)$ $\mathcal{B}_1 \xrightarrow[\delta_Q]{a} \mathcal{B}_2$ существует соответствующая дуга полного автомата $L^\#$.*

Доказательство. Если существует $\mathcal{B}_1 \xrightarrow[\delta_Q]{a} \mathcal{B}_2$, т. е. выполняются условия (2) и (3), то из множества $\alpha(\mathcal{B}_1)$ можно выбрать состояние, которое, не ограничивая общности, мы обозначим A_1. Рассмотрим псевдоблок $A_1\#X_1 \in \mathcal{B}_1$. В полном автомате существуют дуги из

A_1 по буквам соответствующим $A_1\#X_1$ в любое из соответствующих состояний Q_π. Среди этих дуг (дуг выходящих из A_1) можно выбрать дугу $(A_1, a_{{}_{X_1}{}^{A_2}}) = A_2$, где A_2 удовлетворяет первому условию (2) из условий существования дуги автомата $\mathcal{COM}(L)$, т.е. $\delta_\pi(A_1, a) = A_2$. Таким образом мы выбрали соответствующую дугу в автомате $\mathcal{COM}(L)$. □

5. Соответствие дуг базисного автомата дугам полного автомата

Рассмотрим базисный автомат $\mathcal{BA}(L)$ для регулярного языка L (см. [1, 12, 13]).

$$\mathcal{BA}(L) = (\hat{Q}, \Sigma, \hat{\delta}, \hat{S}, \hat{F}\,),$$

Определим базисный автомат $\mathcal{BA}(L)$ следующим образом:

- $\hat{Q}$ – множество пар T = (A,X), таких что $A \in Q_\pi$, $X \in Q_\rho$, приччм A#X. При этом мы будем писать $A = \alpha(T)$ и $X = \beta(T)$;
- функцию $\hat{\delta}$ определим следующим образом. Для всех $T_1, T_2 \in \hat{Q}$ и $a \in \Sigma$ полагаем $\hat{\delta}(T_1, a) \ni T_2$ тогда и только тогда, когда:

  $$\delta_\pi(\alpha(T_1), a) = \{\alpha(T_2)\} \text{ и } \delta_\rho(\beta(T_2), a) = \{\beta(T_1)\}$$

  , т.е. для компоненты α по букве a должен быть переход в $\widetilde{L}$ из T_1 в T_2, и одновременно для компоненты β по букве a должен быть переход в $\widetilde{L^R}$ из T_2 в T_1;
- считаем, что $T \in \hat{S}$ тогда и только тогда, когда $\alpha(T) = s_\pi$;
- аналогично, $T \in \hat{F}$ тогда и только тогда, когда $\beta(T) = s_\rho$.

О п р е д е л е н и е 4. *Введчм* ***раскраску на основе базисного автомата*** $c_\#$ *следующим образом. Для функции*

$$c_\# : \Sigma \times \Sigma_\# \to \mathcal{P}(Q_\pi)$$

полагаем

$$c_\#(a, a_{{}_{X_1}{}^{A_2}}) \ni A_1\,,$$

если в базисном автомате существует переход $\begin{smallmatrix} A_1 \\ X_1 \end{smallmatrix} \xrightarrow[\hat{\delta}]{a} \begin{smallmatrix} A_2 \\ X_2 \end{smallmatrix}$*; иных значений эта функция не содержит.*

Заметим, что, в силу определения базисного автомата

$$c_{\#}(a, a_{X_1 A_2}) \ni A_1 \quad \text{влечёт} \quad \delta_\pi(A_1, a) = A_2. \tag{4}$$

Утверждение 1. *$c_{\#}$ – допустимая раскраска.*

Доказательство. Пусть $A_1 \in c(a, a_{X_2 A_3})$ и $A_1 \in c(a, a_{X_4 A_5})$; тогда $\delta_\pi(A_1, a) = A_3$ и $\delta_\pi(A_1, a) = A_5$ (см. (4)), то есть $A_3 = A_5$. $\square$

Определение 5. *Назовём автомат $K_{\mathcal{J}} = (Q_\pi, \Sigma', \delta_{\mathcal{J}}, S_\pi, F_\pi)$* ***объединением на основе допустимой раскраски*** *c для следующего определения функции $\delta_{\mathcal{J}}$. Пусть $a \in \Sigma'$, и $A_1 \in Q_\pi$. Тогда:*

- *если существует $a_{X_1 A_2} \in \Sigma_{\#}$, такое что $A_1 \in c(a, a_{X_1 A_2})$, то полагаем $\delta_{\mathcal{J}}(A_1, a) = \{A_2\}$;*
- *иначе если $c(a, a_{X_1 A_2}) \neq \varnothing$, то полагаем $\delta_{\mathcal{J}}(A_1, a) = \delta_{\#}(A_1, a_{X_1 A_2})$;*
- *в остальных случаях полагаем $\delta_{\mathcal{J}}(A_1, a) = \varnothing$.*

Это определение корректно, т. к. в силу допустимости раскраски значение $\delta_{\mathcal{J}}$ в первом пункте определено однозначно.

Дуги полного автомата предполагают объединение, если выходные состояния двух дуг совпадают и входные состояния этих же двух дуг совпадают, тогда разные буквы $a_{X_2 A_3}$ и $a_{X_4 A_3}$ из $\Sigma_{\#}$ полного автомата считаем одинаковыми буквами a из Σ для получения исходного канонического автомата $\widetilde{L}$. Потому что мы имея полный автомат и желая построить канонический автомат, не зная базисного, не знаем по какой букве возникло отношение между $A_1 \# X_2$ и $A_1 \# X_4$, но можем предположить, что по одной и той же букве a автомата $\widetilde{L}$.

Не раскрашенные буквы $A_1 \notin c(a, a_{X_1 A_2}) \neq \varnothing$ это отсутствие перехода $\delta_\pi(A_1, a) = A_2$ в автомате $\widetilde{L}$.

Теорема 2. *Объединение на основе раскраски $c_{\#}$ дает канонический автомат.*

Доказательство. Следует из (4). □

Т е о р е м а 3. *Для каждой дуги базисного автомата* $\mathcal{BA}$ $\begin{smallmatrix} A_1 \\ X_1 \end{smallmatrix} \xrightarrow[\hat{\delta}]{a} \begin{smallmatrix} A_2 \\ X_2 \end{smallmatrix}$ *существует соответствующая дуга полного автомата* $L^{\#}$ $A_1 \xrightarrow[\delta_{\#}]{a_{X_1}{}^{A_2}} A_2$.

Доказательство. Предположим для дуги базисного автомата $\begin{smallmatrix} A_1 \\ X_1 \end{smallmatrix} \xrightarrow[\hat{\delta}]{a} \begin{smallmatrix} A_2 \\ X_2 \end{smallmatrix}$ нет дуги $A_1 \xrightarrow[\delta_{\#}]{a_{X_1}{}^{A_2}} A_2$ в полном автомате, т.е. $\delta_{\#}(A_1, a_{X_1}{}^{A_2}) = \varnothing$. Тогда из определения полного автомата получается, что в таблице бинарного отношения $\#$ не должно быть элемента $A_1 \# X_1$. Отсутствие отношения $\#$ между A_1 и X_1 противоречит определению базисного автомата, а значит и сделанному предположению. □

Другими словами в полном автомате, построенном по таблице $\#$, существуют дуги из A_1 по разным буквам соответствующим $A_1 \# X_1$ в любое из состояний соответствующих состояниям Q_π, в каждое по своей букве. Среди этих дуг (дуг выходящих из A_1) можно выбрать дугу $(A_1, a_{X_1}{}^{A_2}) = A_2$ соответствующую действительно существующему переходу $\delta_\pi(A_1, a) = A_2$ в $\widetilde{L}$ и дуге $\delta_\pi(\alpha(\begin{smallmatrix} A_1 \\ X_1 \end{smallmatrix}), a) = \alpha(\begin{smallmatrix} A_2 \\ X_2 \end{smallmatrix})$ в базисном автомате, буквы при этом выбираются из алфавита $\Sigma_{\#}$, а не из исходного Σ.

6. Заключение

Материал статьи предполагает дальнейшее развитие теории построения конструкций walibad, в частности доказательства следующих теорем:

- если для некоторой заданной таблицы бинарного отношения $\#$ существует автомат, для которого покрывающий автомат не эквивалентен исходному автомату, то этот исходный автомат может быть получен переборным способом на основе произвольной помечающей функции для автомата $L^{\#}$ и некоторого допустимого объединения его столбцов (букв);

- для таблицы # некоторого заданного конечного автомата K, обладающего свойством walibad, существует вариант объединения столбцов автомата $L^{\#}$, такой что полученный автомат: во-первых, имеет ту же самую таблицу отношения #; во-вторых, обладает свойством walibad.

Список литературы

[1] *Melnikov B., Melnikova A.* Some properties of the basis finite automaton // The Korean Journal of Computational and Applied Mathematics. 2002. Vol. **9**. №. 1. P. 131–150.

[2] *Долгов В. Н., Мельников Б. Ф.* Построение универсального конечного автомата. I. От теории к практическим алгоритмам // Вестник Воронежского государственного университета. Серия: Физика. Математика. 2013. №. 2. С. 173–181.

[3] *Melnikov B., Melnikova A.* Some more on the basis finite automaton // Acta Informatica, Univ. Sapientiae. 2013. Vol. **5**. №. 2 P. 227–244.

[4] *Lombardy S., Sakarovitch J.* The universal automaton // Logic and Automata. Amsterdam University Press. 2008. P. 457–504.

[5] *Jiang T., Ravikumar B.* Minimal NFA problems are hard // SIAM J. Comput. 1993. Vol. **22**. No. 6. P. 1117–1141.

[6] *Melnikov B.* Once more about the state-minimization of the nondeterministic finite automata // The Korean Journal of Computational and Applied Mathematics. 2000. Vol. **7**. No. 3. P. 655–662.

[7] *Polák L.* Minimalizations of NFA using the universal automaton // Int. J. Found. Comput. Sci. 2005. Vol. **16**. No. 5. P. 999–1010.

[8] *Geldenhuys J., van der Merwe B., van Zijl L.* Reducing nondeterministic finite automata with SAT solvers // Springer. Finite-State Methods and Natural Language Processing. Lecture Notes in Computer Science. 2010. Vol. **6062**. P. 81–92.

[9] *Yo-Sub Han.* State elimination heuristics for short regular expressions // Fundamenta Informaticae. 2013. Vol. **128**. P. 445–462.

[10] *Kameda T., Weiner P.* On the state minimization of nondeterministic finite automata // IEEE Trans. on Comp. 1970. Vol. **19**. No. 7. P. 617–627.

[11] *Долгов В. Н., Мельников Б. Ф.* Об алгоритмах автоматического построения Ватерлоо-подобных конечных автоматов на основе полных автоматов // Эвристические алгоритмы и распределенные вычисления. 2014. Vol **1**. No. 4. С. 24–45.

[12] *Melnikov B.* Once more on the edge-minimization of nondeterministic finite automata and the connected problems // Fundamenta Informaticae. 2010. Vol **104**. No. 3. P. 267–283.

[13] *Мельников Б. Ф.* О звёздной высоте регулярного языка. Часть I: Основные определения // Эвристические алгоритмы и распределённые вычисления. 2014. Т. **1**. №. 1 С. 74–87.

[14] *Мельников Б. Ф.* Недетерминированные конечные автоматы. — Тольятти: Изд-во ТГУ. 2009. 160 с.