

Итерации языков и конечные автоматы

Б. Ф. Мельников, А. А. Вылиток, Е. А. Мельникова

Аннотация—В работе рассмотрено представление итераций конечных языков с помощью специальных недетерминированных конечных автоматов. Для двух итерируемых языков определяется специальное отношение эквивалентности. Приводится сведение условий выполнения этого отношения к определению эквивалентности двух недетерминированных конечных автоматов, построенных на основе двух заданных конечных языков. Также приводится связанное с рассматриваемыми нами вопросами тривиальное доказательство регулярности морфического прообраза регулярного языка. Кроме того, с использованием рассматриваемого отношения эквивалентности формулируются гипотезы о существовании полиномиально-временного алгоритма построения инверсного морфизма, обладающего заданными специальными свойствами, а также о связи рассматриваемых нами проблем с вопросом о возможном равенстве классов $P=NP$.

Ключевые слова—итерации конечных языков, регулярные языки, недетерминированные конечные автоматы.

I. Введение

Настоящую статью можно рассматривать как продолжение [1]. Однако здесь мы не будем заострять вопросы на итерациях бесконечных языков – т.е. будем рассматривать только итерации конечных языков, которые могут быть заданы с помощью различных вариантов конечных автоматов-распознавателей. А связь итераций конечных языков с недетерминированными конечными автоматами будет рассмотрена существенно более подробно, чем в [1].

Ещё в [2] мы впервые рассмотрели бесконечные итерации конечных языков. А именно, мы рассматривали ω -языки вида A^ω , где A – конечный язык над заданным алфавитом Σ . (Мы предполагаем, что $|\Sigma| \geq 2$, а также – если специально не сказано иного – что $A \not\equiv \varepsilon$.) Для двух конечных языков A и B мы рассматривали равенство $A^\omega = B^\omega$, которое равносильно выполнению специального отношения эквивалентности, также определённого в [2]: $A^* \equiv_\infty B^*$ тогда и только тогда, когда

$$\begin{cases} (\forall u \in A^*) (\exists v \in B^*) (u \in \text{Pref}(v)) \\ (\forall v \in B^*) (\exists u \in A^*) (v \in \text{Pref}(u)). \end{cases} \quad (1)$$

Здесь $\text{Pref}(u)$ – множество всех префиксов слова u , включая ε и u . Аналогично для языков:

$$\text{Pref}(A) = \{ u \mid u \in \text{Pref}(v), v \in A \}.$$

В [2] мы рассматривали равенство ω -языков и определённую таким образом эквивалентность в случае произвольных ω -языков и языков соответственно. В настоящей

Статья получена 10 октября 2017.

Борис Феликсович Мельников, Российский государственный социальный университет (email: bf-melnikov@yandex.ru).

Алексей Александрович Вылиток, МГУ им. М. В. Ломоносова (email: vyilitok@cs.msu.su).

Елена Анатольевна Мельникова, Российский государственный социальный университет (email: ya.e.melnikova@yandex.ru).

работе мы будем рассматривать их только для ω -языков типа A^ω и языков типа A^* . Вследствие этого *всюду ниже* в настоящей статье мы будем писать $A \equiv B$ – вместо использовавшегося в [2] обозначения $A^* \equiv_\infty B^*$; итак, условие $A \equiv B$ выполняется тогда и только тогда, когда выполняются оба условия (1).

В [2] были получены необходимые и достаточные условия эквивалентности $A \equiv B$; стоит отметить, что существенно проще эти условия могут быть получены в т.н. префиксном случае, т.е. когда

$$(\forall u, v \in A, u \neq v) (u \notin \text{Pref}(v))$$

поэтому префиксный случай менее интересен. Отметим также, что в большинстве рассматриваемых нами задач менее интересен и случай бесконечных итерируемых языков.¹

Структура настоящей статьи следующая. В разделе II описываются очевидные *достаточные* условия выполнения равенства $A^\omega = B^\omega$; они формулируются в виде трёх шагов построения самих языков A и B .

Разделы III и IV посвящены недетерминированным конечным автоматам. В разделе III приведены обозначения, согласованные с применявшимися в предыдущих работах; например, рассматриваются упрощённые определения функций разметки состояний произвольного недетерминированного автомата состояниями эквивалентного канонического. А в разделе IV приводится несложный алгоритм сведения рассмотренной выше проблемы равенства $A^\omega = B^\omega$ (при $A \not\equiv \varepsilon$ и $B \not\equiv \varepsilon$) к рассмотрению недетерминированных конечных автоматов.

В разделе V приводятся различные варианты формулировки того факта, который выполнялся бы, если бы приведённое ранее, в разделе II, достаточное условие равенства бесконечных итераций *не* являлось бы необходимым и достаточным. На основе этих формулировок, а также описанного в разделе IV алгоритма сведения равенства $A^\omega = B^\omega$ к недетерминированным конечным автоматам, мы в этом же разделе V получаем специальную теорему о канонических конечных автоматах. А на основе *возможных в будущем* доказательств двух утверждений – о возможности/невозможности полиномиально-временной² реализации двух разных алгоритмов проверки равенства $A^\omega = B^\omega$ – мы в заключении, в разделе VI, приводим возможный план доказательства равенства $P=NP$.

¹ Причём можно провести аналогию между рассматриваемыми нами проблемами и т.н. примером М.-П. Шютценберже ([3, гл. 5, пр. 3.3.1]): над *конечным* алфавитом подобных примеров не существует.

Отметим ещё, что приведённая в этой же главе ([3, гл. 5]) теория, относящаяся к максимальным префиксным кодам, связана и с проблемами, рассматриваемыми далее в настоящей статье.

² Употребление термина «полиномиально-временной» в качестве прилагательного-композиата не очень распространено, но представляется авторам весьма удачным. Такое употребление согласовано с терминологией, применяемой Ю. Л. Ершовым, см., например, [4].

II. Достаточное условие равенства бесконечных итераций

Несложно доказываемое достаточное условие равенства бесконечных итераций было приведено в [1], [2] и др.; здесь мы более подробно, чем в предыдущих работах, рассмотрим один из соответствующих примеров.

Итак, *достаточное* условие равенства $A^\omega = B^\omega$ (или определённой нами эквивалентности $A \equiv B$) формулируется в виде следующих 3 шагов.

- 1) Рассматриваем некоторый *новый* конечный алфавит Δ и два произвольных максимальных префиксных кода³ (A_Δ и B_Δ) над этим алфавитом. Для рассматриваемого максимального префиксного кода A мы будем писать $A_\Delta, B_\Delta \in \text{mp}(\Delta)$. То есть всё множество $\text{mp}(\Delta)$ – это множество максимальных префиксных кодов над (конечным) алфавитом Δ .⁴ При этом условие $A_\Delta \equiv B_\Delta$ выполнено – поскольку

$$A_\Delta^\omega = B_\Delta^\omega = \Delta^\omega. \quad (2)$$

∇_1 В качестве примера рассмотрим алфавит

$$\Delta = \{0, 1\}$$

и два таких языка:

$$A_\Delta = \{0, 10, 11\} \quad \text{и} \quad B_\Delta = \{00, 01, 1\}.$$

- 2) Добавляем некоторые слова к языкам A_Δ и B_Δ .⁵ В этом случае для рассматриваемого языка A_Δ мы при этом будем писать

$$A_\Delta, B_\Delta \in \text{mp}^+(\Delta).$$

То есть $\text{mp}^+(\Delta)$ – множество языков над (конечным) алфавитом Δ , каждый из которых в качестве подмножества содержит некоторый максимальный префиксный код над этим алфавитом. При этом эквивалентность $A_\Delta \equiv B_\Delta$ тоже выполняется – также вследствие (2). Важно ещё отметить, что полученные языки, вообще говоря, не являются префиксными.

∇_2 В качестве продолжения предыдущего примера рассмотрим языки

$$A_\Delta = \{0, 10, 11, 1101\} \\ \text{и} \quad B_\Delta = \{00, 01, 1, 111, 1110\} :$$

оба языка получаются из языков предыдущего примера ∇_1 путём добавления конечного числа слов.

- 3) Рассмотрим некоторый морфизм $h : \Delta^* \rightarrow \Sigma^*$; при этом мы допускаем возможность, когда язык

$$C = h(\Delta) = \{h(c) \mid c \in \Delta\}$$

³ Максимальные префиксные коды рассматриваются, например, в [3]. Не будет ошибкой считать, что это название («максимальный префиксный код») – *само* является *определением*: все эти три понятия («код», «максимальный» и «префиксный») по отдельности определены, например, также в [3].

⁴ Это и определённые ниже связанные с ним обозначения полностью согласованы с применявшимися в [2].

⁵ При этом множество добавляемых слов может быть пустым.

Отметим также, что для A_Δ достаточно рассматривать (добавлять) только такие слова $u \in \Delta^*$, для которых $u \notin \text{Pref}(A_\Delta)$; аналогично для B_Δ .

также не является префиксным. Будем рассматривать языки

$$A = h(A_\Delta) \quad \text{и} \quad B = h(B_\Delta)$$

и для вышеуказанного языка C писать

$$A, B \in \text{mp}^+(C).$$

При этом если, например, $A_\Delta \in \text{mp}(\Delta)$, то будем писать $A \in \text{mp}(C)$. Для полученных таким образом языков условие $A \equiv B$ также выполняется.

∇_3 Продолжим рассматривать предыдущий пример ∇_2 . Для (непрефиксного) морфизма

$$h(0) = ab, \quad h(1) = aba$$

мы получаем следующие языки:

$$A = \{ab, abaab, abaaba, abaabaababa\}$$

и

$$B = \{abab, ababa, aba, abaabaaba, abaabaabaab\}.$$

Итак, очевидное достаточное условие равенства $A^\omega = B^\omega$ таково: если языки A и B получены описываем здесь способом, то $A^\omega = B^\omega$ (т.е. $A \equiv B$).⁶

III. Недетерминированные конечные автоматы – применяемые обозначения

Будем использовать следующие обозначения (они согласованы с [8], [9]). Пусть

$$K = (Q, \Sigma, \delta, S, F) - \quad (3)$$

некоторый автомат (недетерминированный конечный автомат Рабина-Скотта), определяющий регулярный язык $L = \mathcal{L}(K)$. Здесь Q – множество состояний, а $S \subseteq Q$ и $F \subseteq Q$ – соответственно множества входных и выходных состояний. Функция переходов δ автомата (3) рассматривается как

$$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow \mathcal{P}(Q),$$

где запись $\mathcal{P}(Q)$ обозначает множество всех подмножеств Q ; отметим отличие от вида функции переходов, применявшейся в [9].

Входные и выходные языки состояния $q \in Q$, т.е. языки, определяемые автоматами

$$(Q, \Sigma, \delta, S, \{q\}) \quad \text{и} \quad (Q, \Sigma, \delta, \{q\}, F)$$

соответственно, обозначаются $\mathcal{L}_K^{\text{in}}(q)$ и $\mathcal{L}_K^{\text{out}}(q)$.

Для автомата (3) и пары его состояний $p, r \in Q$ будем использовать следующие обозначения. Если $\delta(p, a) \ni r$, то будем писать $p \xrightarrow{a} r$ или $p \xrightarrow{a}_K r$. Будем иногда опускать индексы δ и/или K – когда автомат подразумевается. Аналогичные обозначения будут использованы и для слов. Например, мы будем писать $p \xrightarrow{u} r$ – если существует путь с пометкой $u \in \Sigma^*$ из состояния p в состояние r ; при этом p может быть опущено – если имеется в виду путь из (какого-либо) входа, и т.п.

⁶ Возможные варианты *применения* этого условия в других задачах теории формальных языков (в том числе – возможное применение в системах автоматизации построения компиляторов) описано в работах [6], [7].

Зеркальный автомат для автомата (3), т.е. автомат $(Q, \Sigma, \delta^R, F, S)$, где

$$q' \xrightarrow{\delta^R} q'' \quad \text{тогда и только тогда, когда} \quad q'' \xrightarrow{\delta} q',$$

будет обозначаться K^R ; заметим, что K^R всегда определяет язык L^R (зеркальный образ языка L).

Для рассматриваемого языка L его канонический автомат будем обозначать записью \tilde{L} .⁷ Элементы пятёрок, определяющих автоматы \tilde{L} и \tilde{L}^R для заданного регулярного языка L , будем обозначать следующим образом:

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

Аналогично [8], [9], мы будем использовать две специальные функции (φ^{in} и φ^{out}), вида

$$\varphi^{in} : Q \rightarrow \mathcal{P}(Q_\pi) \quad \text{и} \quad \varphi^{out} : Q \rightarrow \mathcal{P}(Q_\rho) \quad -$$

т.н. функции разметки состояний автомата (3). В настоящей статье достаточно привести только краткие определения этих функций:

$$\varphi_K^{in}(q) \ni A \quad \text{тогда и только тогда, когда} \\ (\exists u \in \Sigma^*) \left(\frac{u}{K} \rightarrow q \ \& \ \frac{u}{\tilde{L}} \rightarrow A \right);$$

аналогично

$$\varphi_K^{out}(q) \ni X \quad \text{тогда и только тогда, когда} \\ (\exists v \in \Sigma^*) \left(\frac{v}{K^R} \rightarrow q \ \& \ \frac{v}{\tilde{L}^R} \rightarrow X \right).$$

Заметим, что такое определение не является конструктивным – однако мы можем описать и алгоритм построения этих функций; один из возможных алгоритмов приведён в [8].⁸

IV. Итерации языков

и недетерминированные конечные автоматы

В этом разделе мы рассмотрим несложный алгоритм сведения рассмотренной выше проблемы равенства ω -языков вида A^ω (где $A \not\ni \varepsilon$) к конечным автоматам. С их помощью мы, во-первых, мы опишем простой алгоритм проверки равенства бесконечных итераций конечных языков (т.е. проверки выполнения условия (1)); этот алгоритм мы будем использовать в дальнейших построениях настоящей статьи. А во-вторых, мы сформулируем некоторые следствия такой гипотезы: сформулированное в разделе II достаточное условие равенства бесконечных итераций *не* является необходимым и достаточным.

Для языка $A = \{u_1, \dots, u_n\}$, где $u_i = a_{i,1} \dots a_{i,l_i}$ для $i = 1, \dots, n$, рассмотрим недетерминированный конечный автомат $\mathcal{K}(A) = (Q, \Sigma, \delta, \{s\}, Q)$, где

$$Q = \{s\} \cup \bigcup_{i \leq n, j < l_i} \{q_{i,j}\},$$

⁷ Аналогично [8], мы считаем каноническим автоматом детерминированный автомат, содержащий минимально возможное число состояний. При этом, также аналогично [8], мы *не* требуем «всюдоопределённости» этого автомата – и, следовательно, не включаем в него возможное «дохлое» состояние («dead state»).

⁸ В настоящей статье мы практически не рассматриваем вопросы, связанные со сложностью алгоритмов. Однако в данном случае отметим, что построения функций φ^{in} и φ^{out} не требуют дополнительных вычислений: согласно приведённому в [8] алгоритму, они могут быть построены в процессе детерминизации (и последующей «канонизации») заданного автомата.

со следующей функцией переходов δ (мы рассматриваем приведённые далее определения для каждого $i = 1, \dots, n$). Если $|u_i| = 1$, то $\delta(s, a_{i,1}) \ni s$; иначе⁹

$$\delta(s, a_{i,1}) \ni q_{i,1}, \quad \delta(q_{i,l_i-1}, a_{i,l_i}) = \{s\} \\ \text{и} \quad \delta(q_{i,j-1}, a_{i,j}) = \{q_{i,j}\} \\ \text{для каждого возможного} \quad j \leq l_i - 1.$$

Очевидно, что такой автомат определяет язык $\text{Pref}(A^*)$; поэтому условие (1) можно проверить, ответив на вопрос об эквивалентности автоматов $\mathcal{K}(A)$ и $\mathcal{K}(B)$.

Для ответа на поставленный вопрос мы можем использовать канонические автоматы, эквивалентные $\mathcal{K}(A)$ и $\mathcal{K}(B)$ и определённые согласно [8]. Отметим ещё, что *каждое* состояние соответствующего канонического автомата соответствует некоторому подмножеству множества состояний рассматриваемого автомата – например, автомата $\mathcal{K}(A)$.

Очевидно, что в качестве примера сложно рассматривать автоматы, построенные для интересных примеров языков из раздела II – поскольку, например, автомат $\mathcal{K}(A)$ для языка

$$A = \{ab, aba\ ab, aba\ aba, aba\ aba\ ab\ aba\}$$

содержит следующее число состояний:

$$1 + (|ab| - 1) + (|aba\ ab| - 1) + (|aba\ aba| - 1) + \\ (|aba\ aba\ ab\ aba| - 1) = \\ 1 + (2 - 1) + (5 - 1) + (6 - 1) + (11 - 1) = 21.$$

Это достаточно много (пример получается громоздким), поэтому рассмотрим автомат $\mathcal{K}(D)$ только для языка $D = \{ab, aba\}$; даже этот небольшой пример достаточно информативен. Точнее, мы рассмотрим два следующих автомата: один – в точности совпадающий с определённым ранее, рис. 1:

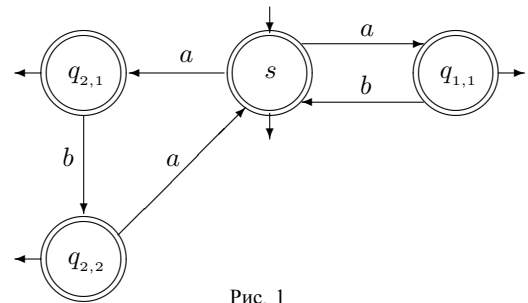


Рис. 1

и второй, полученный из него с помощью очевидных эквивалентных преобразований, он приведён на рис. 2:

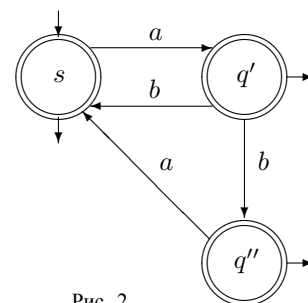


Рис. 2

⁹ Заметим, что мы уже предположили, что $A \not\ni \varepsilon$, поэтому $|u_i| \geq 1$.

(Здесь мы обозначаем выходные состояния *не* двойными кружками, а пользуемся другой, более естественной системой обозначений, схожей с принятой, например, в [10] и употребляемой в большинстве современных публикаций: входные состояния помечаются малой входной стрелкой, а выходные – малой выходной. Такая система обозначения, как уже было отмечено, использовалась нами и в [8].)

Эквивалентный им канонический автомат приведён на следующем рис. 3:

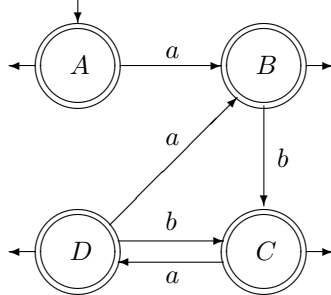


Рис. 3

Как мы уже отмечали, мы можем использовать два таких канонических автомата для определения эквивалентности автоматов $\mathcal{K}(A)$ и $\mathcal{K}(B)$ – а, следовательно, и для ответа на вопрос, выполнено ли условие (1). Однако основная причина рассмотрения подобных автоматов – такая.

В статье [2] было показано, что описанные выше (раздел II) достаточные условия равенства бесконечных итераций являются также необходимыми – об этом будет подробнее сказано ниже. В нескольких работах одного из авторов настоящей статьи, опубликованных до статьи [2] (например, в [11]), мы не могли использовать факт необходимости сформулированных условий; однако в этих статьях были получены различные утверждения об условиях равенства некоторых подклассов класса КС-языков. Примерно так мы поступим и в настоящей статье: предполагая возможность невыполнения таких необходимых условий, мы сформулируем гипотезу о существовании недетерминированных конечных автоматов специального вида – и далее (на основе результатов [2]) покажем, что подобных автоматов не существует.

Итак, сначала отметим следующие два утверждения. Во-первых, в [2] было также доказано следующее. Отрицание того, что

достаточные условия равенства бесконечных итераций, сформулированные в разделе II, являются необходимыми и достаточными,

может быть эквивалентно переформулировано следующим образом:

существуют некоторый (конечный) язык D и (для него) некоторое слово $v \notin D^$, такие что $D \equiv D \cup \{v\}$.*

Такой язык D может быть назван «минимальным».

А во-вторых, согласно определению¹⁰ функции φ^{in} , каждое состояние канонического автомата соответствует некоторому подмножеству состояний автомата, ему эквивалентного; например, это верно для автомата $\mathcal{K}(A)$

и эквивалентного ему канонического автомата.¹¹ Это соответствие означает следующее. Для некоторого состояния \tilde{q} канонического автомата и некоторого слова v из входного языка этого состояния (т.е. $\mathcal{L}^{in}(\tilde{q})$), существуют некоторые состояния q_1, \dots, q_m автомата $\mathcal{K}(A)$, такие что

$$v \in \mathcal{L}^{in}(q_1), \dots, v \in \mathcal{L}^{in}(q_m), \quad 12$$

$$q_i \neq s \quad \text{для } i = 1, \dots, m;$$

Поэтому для некоторого «минимального» языка D и некоторого слова $v \notin D^*$ выполнено условие

$$D \equiv D \cup \{v\},$$

а также

$$\bigcup_{i \leq m} \mathcal{L}^{out}(q_i) \supseteq \text{Pref}(D^*).$$

Мы можем рассматривать эту «минимальность» в определённом выше смысле – однако её для рассматриваемого нами языка A (именно для него выше был построен автомат $\mathcal{K}(A)$) можно определять и другими способами.

Например, это можно сделать следующим образом. На множестве всех конечных языков B , для которых для рассматриваемого A выполнено условие (1), вводится некоторая «естественная» метрика – и среди всех таких языков выбирается язык, являющийся по введённой метрике минимальным – или один из таких языков. А в качестве такой «естественной» метрики можно рассматривать, например, длину максимального слова в языке. (Отметим, что в [1] было приведено обобщение подобной метрики на случай бесконечных языков.)

Ещё заметим, что рассмотрение различных подобных метрик может быть применено для описания различных прикладных алгебраических моделей – как рассматривавшихся в [12], так и связанных с теорией, рассмотренной нами в [13].

V. Необходимое условие равенства бесконечных итераций

На основании вышеизложенного мы можем следующим образом переформулировать тот факт, что приведённое в разделе II достаточное условие равенства бесконечных итераций *не* является необходимым и достаточным:

существует некоторый конечный язык D , такой что у канонического конечного автомата, эквивалентного $\mathcal{K}(D)$,¹³ существует некоторое состояние \tilde{q} , не являющееся входным, такое что

$$\mathcal{L}^{in}(\tilde{q}) \cap D^* = \emptyset \quad \text{и} \quad \mathcal{L}^{out}(\tilde{q}) \supseteq \text{Pref}(D^*).$$

Заметим, что в рассмотренном примере канонического автомата (рис. 3) существует единственное состояние (B), для которого $\mathcal{L}^{in}(B) \cap D^* = \emptyset$. Очевидно, что условие $\mathcal{L}^{out}(B) \supseteq \text{Pref}(D^*)$ для этого состояния не выполняется – поскольку $\mathcal{L}^{out}(B) \not\supseteq a$.

¹¹ Фактически это соответствие описывается как $(\varphi^{in})^{-1}$.

¹² Таких состояний в построенном нами автомате $\mathcal{K}(A)$, вообще говоря, больше 1.

¹³ Иными словами – для канонического конечного автомата, определяющего регулярный язык $\text{Pref}(D^*)$.

¹⁰ Кратко – см. раздел III. Подробнее – см. [8] и др.

Описанное соответствие также может быть построено на основе [8]. Фактически им является рассматриваемая в той статье функция $(\varphi^{in})^{-1}$. Это соответствие означает следующее. Для некоторых состояний \tilde{q} канонического автомата и слова v входного языка этого состояния (т.е. слова языка $\mathcal{L}^{in}(\tilde{q})$) существуют некоторые состояния q_1, \dots, q_m автомата $\mathcal{K}(A)$, такие что

$$v \in \mathcal{L}^{in}(q_1), \dots, v \in \mathcal{L}^{in}(q_m).$$

Поэтому если для «минимального» языка D и некоторого слова $v \notin D^*$ выполнено условие $D \equiv D \cup \{v\}$, то

$$q_i \neq s \quad \text{для } i = 1, \dots, m,$$

а также

$$\bigcup_{i \leq m} \mathcal{L}^{out}(q_i) \supseteq \text{Pref}(D^*).$$

И наоборот – тот факт, что сформулированное в разделе II достаточное условие равенства бесконечных итераций является необходимым и достаточным, равносильно несуществованию подобного «минимального» языка D .

А основное доказанное в [2] утверждение ([2, Th. 1]) сформулировано следующим образом. Для некоторой пары конечных языков A и B , таких что $A \equiv B$, существует конечный язык D (пусть $D = \{u_1, \dots, u_n\}$), для которого выполнено такое условие. Для некоторого нового алфавита $\Delta = \{c_1, \dots, c_n\}$ существуют два языка $A', B' \subseteq \Delta^*$, такие что:

- оба языка (A' и B') в качестве подмножеств содержат максимальные префиксные коды над алфавитом Δ ; ¹⁴
- для морфизма $h : \Delta^* \rightarrow \Sigma^*$, где

$$h(c_1) = u_1, \dots, h(c_n) = u_n,$$

выполнены условия $h(A') = A$ и $h(B') = B$.

Итак, достаточное условие равенства бесконечных итераций является необходимым и достаточным. Вследствие этого выполняется такая теорема о канонических конечных автоматах.

Теорема 1: Для любого конечного языка D у канонического конечного автомата, определяющего регулярный язык $\text{Pref}(D^*)$, не существует состояния \tilde{q} (отличного от входного), такого что

$$\mathcal{L}^{in}(\tilde{q}) \cap D^* = \emptyset \quad \text{и} \quad \mathcal{L}^{out}(\tilde{q}) \supseteq \text{Pref}(D^*). \quad \square$$

В качестве важного комментария к материалу настоящего раздела упомянем следующий давно известный, но совсем не тривиальный факт: если язык B регулярен ¹⁵, то для любого морфизма $h : \Delta^* \rightarrow \Sigma^*$ регулярен и язык $h^{-1}(B)$. ¹⁶ Специально подчеркнём, что здесь язык B никак не связан с морфизмом h — именно из этого и следует нетривиальность данного факта. При этом важно отметить, что, несмотря на то, что для любого $u \in \Delta^*$ запись $h(u)$ определяет слово ([14]), для любого $v \in \Sigma^*$ запись $h^{-1}(v)$ определяет множество слов (язык) над алфавитом Δ :

$$h^{-1}(v) = \{u \in \Delta^* \mid h(u) = v\};$$

¹⁴ Эти коды, вообще говоря, различны.

¹⁵ При этом очевидно, что регулярен и часто используемый нами язык-итерация A^* .

¹⁶ Одно из имеющихся в литературе названий этого утверждения – «задача Ж.-Э. Пина».

при этом определяемое множество может быть и пустым. А язык $h^{-1}(B)$ для любого $B \subseteq \Sigma^*$ строго определяется следующим образом:

$$h^{-1}(B) = \bigcup_{v \in B} h^{-1}(v)$$

(и также может быть пустым). Кроме того, отметим, что оба имеющиеся по ссылке [15] доказательства последнего утверждения ¹⁷ вряд ли интересны для построений, приведённых в нашей статье – по следующим двум причинам.

- В доказательстве, озаглавленном там как «короткое алгебраическое» (“Short algebraic proof”), используется моноид, определяющий заданный регулярный язык; однако (согласно, например, [16], [17]) существует целый класс таких моноидов ¹⁸ – что существенно усложняет интересующие нас в первую очередь алгоритмические вопросы подобных построений.
- А в приведённом там доказательстве «с помощью конечных автоматов» ¹⁹ используются только детерминированные автоматы – что, по мнению авторов, делает процесс доказательства существенно более сложным.

Итак, пусть регулярный язык B задан; рассмотрим произвольный определяющий его конечный автомат

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

(здесь не имеет значения, является ли автомат K детерминированным, и, в случае недетерминированности, разрешаем ли мы ε -переходы). Далее рассмотрим автомат (с тем же самым множеством состояний, в том числе стартовых и финальных)

$$K_\Delta = (Q, \Delta, \delta_\Delta, S, F),$$

где функция δ_Δ определяется следующим образом: $\delta_\Delta(q, a) \ni q'$ (для некоторых $q, q' \in Q$ и $a \in \Delta$) тогда и только тогда, когда язык автомата

$$K_{q,q'} = (Q, \Sigma, \delta, \{q\}, \{q'\})$$

содержит слово $h(a)$. (Заметим, что последний автомат, вообще говоря, не является детерминированным, т.к. $\delta_\Delta(q, a)$ определяется неоднозначно.) Или, применяя другую систему обозначений, условие $\delta_\Delta(q, a) \ni q'$ выполнено тогда и только тогда, когда

$$p \xrightarrow[h(a)]{K} r.$$

При этом очевидно, что автомат K_Δ определяет некоторое слово (над алфавитом Δ) тогда и только тогда, когда морфический образ этого слова определяется автоматом K (последний при принятии этого слова может оказаться в одном из состояний множества F) – и поэтому автомат K_Δ определяет язык $h^{-1}(B)$, что и требовалось доказать.

¹⁷ Приведённые на этом сайте в момент подготовки настоящей статьи – 10 октября 2017 г.

¹⁸ В литературе обычно используется связь полугрупп с конечными автоматами Мили. Однако на её основе может быть очень легко получена и интересующая нас значительно больше связь полугрупп с конечными автоматами Рабина–Скотта (автоматами Медведева).

¹⁹ В нём используются автоматы Рабина–Скотта.

VI. Заключение

Вследствие изложенного выше, можно сформулировать такой план возможного доказательства равенства $P=NP$; этот план улучшает формулировку, приведённую в [1]. А именно, для доказательства этого равенства было бы достаточно доказать следующие два утверждения.

- 1) Проблема проверки неэквивалентности определённых выше конечных автоматов $\mathcal{K}(A)$ и $\mathcal{K}(B)$ (для заданных конечных языков A и B) принадлежит классу NP.
- 2) Существует полиномиально-временной алгоритм «построения инверсного морфизма». То есть для заданного конечного языка $A \in \Sigma^*$ можно за полиномиальное время получить «минимальный» (в сформулированном выше смысле) язык D , такой что $A \equiv D$.

Небольшие комментарии – сначала для первой из этих гипотез. По-видимому, его доказательство может быть получено аналогично доказательству проблемы AL1 (а, возможно, и AL9) из классической монографии [18].

Перейдём к комментариям для второй гипотезы. В настоящее время для неё у авторов нет ни необходимого алгоритма, ни доказательства его несуществования. Но в качестве аргумента существования подобного алгоритма (алгоритма полиномиально-временного построения такого языка D) можно привести следующий. Для каждого слова $w \in D$ существует зависящее от этого слова натуральное число (пусть $n(w)$), такое что $w^{n(w)} \in A$; при этом для каждого слова $u \in A$ все слова w , такие что для некоторого $n \in \mathbb{N}$ выполнено равенство $w^n = u$, строятся за полиномиальное время.

Также стоит отметить следующее. Конечно же, не является полиномиально-временным алгоритм построения всех вариантов представления заданного слова u в виде конкатенации слов некоторого заданного конечного множества A ; тривиальный пример –

$$u = a^n, \quad A = \{a, a^2, \dots, a^{n-1}\},$$

для него мы имеем полиномиальный (в зависимости от n) размер задачи, экспоненциальное (тоже в зависимости от n) число вариантов представления – и, следовательно, экспоненциальное (в зависимости от размера задачи) число вариантов представления. Однако из последнего факта не следует несуществование полиномиально-временного алгоритма для *всей* рассматриваемой нами задачи – «построения инверсного морфизма».

Ещё отметим, что доказательство подзадачи этой задачи (в терминологии этого раздела – «второго утверждения») мы собираемся рассмотреть в нашей следующей публикации. А именно – мы собираемся рассмотреть упрощённый вариант задачи построения общего инверсного морфизма для двух заданных конечных языков A и B , таких что $A \equiv B$.

Также в качестве возможного в будущем применения рассмотренной здесь теории в других задачах теории формальных языков отметим, во-первых, задачи, связанные с кратко упомянутыми выше ([6], [7], [12]) – в них, по-видимому, можно будет получить аналогичные результаты в более широких подклассах класса контекстно-свободных языков, и, во-вторых, задачи разработки алгоритмов эквивалентного преобразования

конечных автоматов, связанных с алгоритмами, описанными в [9], [19].

Список литературы

- [1] Алексеева А. Г. Мельников Б. Ф. *Итерации конечных и бесконечных языков и недетерминированные конечные автоматы*. Вектор науки Тольяттинского государственного университета. № 3. 2011. С. 30–33.
- [2] Melnikov B. *The equality condition for infinite concatenations of two sets of finite words*. International Journal of Foundations of Computer Science. Vol. 4. No. 3. 1993. P. 267–274.
- [3] Lallement G. *Semigroups and Combinatorial Applications*. N.Y.: John Wiley & Sons, Inc., 1979, 376 p. (Лаллеман Ж. *Полугруппы и комбинаторные приложения*. М.: Мир, 1985, 437 с.)
- [4] Целищев В. В., Ершов Ю. Л. *Алгоритмы и вычислимость в человеческом познании*. Новосибирск: Издательство СО РАН, 2012, 505 с.
- [5] Мельников Б. Ф. *Некоторые следствия условия эквивалентности однозначных скобочных грамматик*. Вестник Московского университета, сер. 15. № 3. 1991. С. 51–53.
- [6] Дубасова О. А., Мельников Б. Ф. *Об одном расширении класса контекстно-свободных языков*. Программирование (РАН). № 6. 1995. С. 46–58.
- [7] Melnikov B., Kashlakova E. *Some grammatical structures of programming languages as simple bracketed languages*. Informatica (Lithuanian Acad. of Sci.). Vol. 11. No. 4. 2000. P. 441–454.
- [8] Melnikov B. *Once more on the edge-minimization of nondeterministic finite automata and the connected problems*. Fundamenta Informaticae. Vol. 104. No. 3. 2010. P. 267–283.
- [9] Melnikov B. *The complete finite automaton*. International Journal of Open Information Technologies. Vol. 5. No. 10. 2017. P. 9–17.
- [10] Brauer W. *Automatentheorie. Eine Einführung in die Theorie endlicher Automaten*. Stuttgart: B. G. Teubner, 1984, 493 S. (Брауэр В. *Введение в теорию конечных автоматов*. М.: Радио и связь, 1987, 391 с.)
- [11] Мельников Б. Ф. *Некоторые следствия условия эквивалентности однозначных скобочных грамматик*. Вестник Московского университета, сер. 15. № 3. 1991. С. 51–53.
- [12] Мельников Б. Ф. *Описание специальных подмоноидов глобального надмоноида свободного моноида*. Известия высших учебных заведений. Математика. № 3. 2004. С. 46–56.
- [13] Данг В. В., Корабельщикова С. Ю., Мельников Б. Ф. *О задаче нахождения минимальной полугруппы аппроксимации*. Известия высших учебных заведений. Поволжский регион. Физико-математические науки. № 3 (35). 2015. С. 88–99.
- [14] Salomaa A. *Jewels of Formal Language Theory*. Rockville (Maryland): Computer Science Press, Inc. 1981, 144 p. (Саломая А. *Жемчужины теории формальных языков*. М.: Мир, 1986, 162 с.)
- [15] Computer Science Stack Exchange [Электронный ресурс] = Обмен знаниями о теоретической информатике. – Режим доступа: <https://cs.stackexchange.com/questions/14785>, свободный. – Загл. с экрана, англ. яз.
- [16] Глушков В. М. *Абстрактная теория автоматов*. Успехи математических наук. Т. 16. № 6 (101). 1961. С. 3–62.
- [17] Скорняков Л. А. (ред). *Общая алгебра. Т. 2*. М.: Наука, 1991, 480 с.
- [18] Garey M., Johnson D.S. *Computers and Intractability*. USA: W. H. Freeman and Company. 1979, 338 p. (Гэри М., Джонсон Д. *Вычислительные машины и труднорешаемые задачи*. М.: Мир, 1982, 416 с.)
- [19] Мельников Б. Ф., Сайфуллина М. Р. *О некоторых алгоритмах эквивалентного преобразования недетерминированных конечных автоматов*. Известия высших учебных заведений. Математика. № 4. 2009. С. 67–72.

Iterations of languages and finite automata

Boris Melnikov, Alexey Vylitok, Elena Melnikova

Abstract—In this paper we consider the representation of iterations of finite languages using special nondeterministic finite automata. For two iterated languages, a special equivalence relation is defined. We give the reduction of the conditions for the fulfillment of this relation to the definition of the equivalence of two nondeterministic finite automata, based on two given finite languages. Also we are considering a trivial proof of the regularity of the morphic preimage of regular language; this proof is related to the issues. In addition, using the equivalence relation under consideration hypotheses are formulated on the existence of a polynomial-time algorithm for the construction of an inverse morphism, possessing the specified special properties, as well as the relationship of the problems we are considering with the question of the possible equality of classes $P=NP$.

Keywords—iterations of finite languages, regular languages, nondeterministic finite automata.