Применение лепестковых конечных автоматов для проверки выполнения частного случая гипотезы Zyu (для заданного конечного языка)

Б. Ф. Мельников, А. А. Мельникова

***Аннотация*—В статье мы продолжаем исследование свойств специального бинарного отношения эквивалентности в бесконечности. Ранее мы описали одну гипотезу теории формальных языков, названную нами гипотезой Zyu; одна из нескольких её эквивалентных формулировок может быть выражена так. Для двух конечных и не содержащих пустого слова языков A и B мы можем следующим образом записать необходимое и достаточное условие того, что итерация любого из этих языков принадлежит множеству префиксов второго языка: существует некоторый алфавит (отличный от алфавита, над которым заданы A и B), над ним – два некоторых максимальных префиксных кода (вообще говоря, различных) и два языка, содержащих эти максимальные префиксные коды как подмножества, а также некоторый морфизм между двумя рассматриваемыми алфавитами. Тогда, применяя к последним двум языкам этот морфизм, мы получаем исходные языки A и B.**

Мы рассматриваем эквивалентную формулировку этой гипотезы, использующую не два языка над исходным алфавитом, а только один; при этом важно отметить, что такой вариант гипотезы Zyu заключается в выполнении некоторого условия для любого конечного языка. Используя приводимую формулировку, предмет настоящей статьи можно кратко описать следующим образом: как подобный вариант гипотезы Zyu проверить не для любого конечного языка, а для одного заданного конкретного языка A. При этом мы не стремимся к полиномиальности алгоритма такой проверки: из такой полиномиальности следовала бы и полиномиальность алгоритма проверки эквивалентности двух заданных недетерминированных конечных автоматов.

В статье приводятся формулировки варианта гипотезы Zyu, использующие лепестковый недетерминированный автомат, построенный по некоторому заданному конечному языку A. После этого приводится формулировка ещё одного варианта этой гипотезы – использующая канонический автомат, эквивалентный такому лепестковому автомату. Все варианты гипотезы иллюстрируются примерами, в которых рассматривается один и тот же язык; при этом, конечно, сами примеры приводятся только для одного языка – в то время как гипотезы формулируются для всего множества конечных языков.

***Ключевые слова*—формальные языки, итерации языков, морфизмы, алгоритмы, недетерминированные конечные автоматы, лепестковые автоматы, процедура детерминизации, гипотеза Zyu.**

I. ВВЕДЕНИЕ

В статье мы продолжаем тематику публикаций [1], [2], [3], [4] и др. Во всех упомянутых статьях мы исследовали свойства специального бинарного отношения $\Diamond\!\!\!\!\!-$, впервые рассмотренного нами (в несколько иной интерпретации) ещё в 1993 г., в [5], – т. н. отношения эквивалентности в бесконечности.

Впоследствии – в [6], [7], [8] – мы описали одну гипотезу теории формальных языков, названную нами гипотезой (Ⱬ), или, если без формул, то “Zyu”; одна из нескольких её эквивалентных формулировок может быть выражена так. Для двух конечных и не содержащих пустого слова ε языков (A и B) мы можем записать необходимое и достаточное условие того, что итерация любого из этих языков принадлежит множеству префиксов другого языка, – причём записать его следующим образом. Существует некоторый алфавит Δ (отличный от алфавита Σ, над которым заданы A и B), над ним – два максимальных префиксных кода (вообще говоря, различных) и два языка (A_Δ и B_Δ), содержащих эти максимальные префиксные коды как подмножества, а также некоторый морфизм $h : \Delta^* \to \Sigma^*$. При этом исходные языки A и B образуются путём применения этого морфизма к языкам A_Δ и B_Δ.

Кроме того, мы в упомянутых работах (также см. [9]) показали связь этой гипотезы с возможной переформулировкой гипотезы (равенства) $\mathrm{P}=\mathrm{NP}$: выполнение гипотезы (Ⱬ) влечёт выполнение этого равенства. Кратко план такого сведéния, приведённый в [9], может быть записан следующим образом. По некоторому недетерминированному конечному автомату специальным образом строится пара конечных языков – и для этой пары показывается, что если *бы* была выполнена гипотеза (Ⱬ), то существовал бы и алгоритм проверки выполнения отношения эквивалентности в бесконечности за полиномиальное время. Но, с другой стороны, взяв произвольный недетерминированный конечный автомат и рассматривая его как определённый в наших предыдущих публикациях автомат NSPRI [3], мы бы за полиномиальное время могли бы построить соответствующую этому автомату пару конечных языков; а эта пара удовлетворяет отношению эквивалентности в бесконечности тогда и только тогда, когда язык автомата NSPRI совпадает с универсальным языком над заданным алфавитом (т. е. языком, содержащим все возможные конечные слова). Таким образом, осуществив кратко описанный здесь план, мы тем самым покажем, что выполнение вышеупомянутой гипотезы (Ⱬ) влечёт выполнение равенства $\mathrm{P}=\mathrm{NP}$.

Существует эквивалентная формулировка рассматриваемой в статье гипотезы (Ⱬ) – формулировка, использующая не два языка над исходным алфавитом Σ (языки

Статья получена 19 декабря 2022 г.

Борис Феликсович Мельников, Университет МГУ – ППИ в Шэньчжэне (bormel@smbu.edu.cn).

Александра Александровна Мельникова, Димитровградский инженерно-технологический институт – филиал Национального исследовательского ядерного университета «МИФИ» (super-avahi@yandex.ru).

A и B), а только один язык (подробности далее); но при этом важно отметить, что, конечно, гипотеза ($\mathfrak{A}$) заключается в выполнении некоторого условия *для любого* конечного языка. И, используя упомянутую эквивалентную формулировку, предмет настоящей статьи можно кратко описать следующим образом: как подобный вариант гипотезы ($\mathfrak{A}$) проверить *для одного* заданного конкретного конечного языка A. При этом мы не стремимся к полиномиальности алгоритма такой проверки: из подобной полиномиальности следовала *бы* и полиномиальность алгоритма проверки эквивалентности двух заданных недетерминированных конечных автоматов (а также выполнение равенства $P = NP$).

Приведём содержание статьи по разделам.

В разделе II приведены предварительные сведения и описаны основные обозначения. Даются ссылки на предыдущие статьи авторов, в которых приведены общие обозначения, связанные с конечными автоматами, и, в частности, с т. н. лепестковыми автоматами. Описываются несколько эквивалентных формулировок гипотезы ($\mathfrak{A}$), также приведённых в предыдущих публикациях.

В конце раздела II приведено описание конкретного варианта процедуры детерминизации: такая конкретизация нужна потому, что мы впоследствии (в разделе VI) воспользуемся каноническими автоматами для очередной эквивалентной формулировки гипотезы ($\mathfrak{A}$).

В отдельный раздел III вынесены подробные пояснения к двум рисункам (2 и 3), описывающим алгоритм построения функции Ф – несмотря на то, что всё это также можно было бы отнести к предварительным сведениям. Функция Ф необходима для нескольких формулировок гипотезы ($\mathfrak{A}$). Отметим, что приведённые рисунки являются существенным изменением схожих рисунков, которые поясняли связанные понятия, а именно – экспоненциальный по времени алгоритм проверки выполнения условия $A \lessdot\!\!\!> B$, см. [1], [2] и намного более раннюю публикацию [10] [1].

В разделе IV вводится определение применяемого в настоящей статье варианта лепестковых автоматов (автоматов $\mathcal{K}(A)$ для заданного конечного языка A). Отметим по этому поводу, что в наших предыдущих работах применялись по крайней мере три разных варианта их определения (варианта автомата, обозначавшегося $\mathcal{K}(A)$):

- с одним выходным состоянием («главным»);
- с каждым состоянием, являющимся выходным (как в настоящей статье);
- как ω-автомат, [11].

В частности, в разделе IV приводится пример лепесткового автомата для языка, также рассматривавшегося нами ранее в нескольких предыдущих статьях.

В разделе V приводится формулировка варианта гипотезы ($\mathfrak{A}$), использующая недетерминированный автомат $\mathcal{K}(A)$ для заданного конечного языка A. После этого, в разделе VI, приводится формулировка ещё одного варианта гипотезы ($\mathfrak{A}$) – использующая канонический автомат, эквивалентный автомату $\mathcal{K}(A)$. В обоих этих разделах, V и VI, продолжается рассмотрение примера, начатого в разделе IV.

[1] В публикации 1996 г. подобных рисунков не было; в связи с этим та публикация, по-видимому, сложна для понимания – несмотря на то, что объём комментариев к приведённому в ней алгоритму большой.

Раздел VII – заключение; в нём мы кратко формулируем направления дальнейшей работы, связанные с вопросами, рассматриваемыми в настоящей статье.

II. ПРЕДВАРИТЕЛЬНЫЕ СВЕ́ДЕНИЯ

A. Общие определения и обозначения

Необходимые предварительные сведения про недетерминированные конечные автоматы – причём очень подробные – были приведены в трёх разделах недавно опубликованных статей [1], [2]; поэтому здесь эту информацию мы повторять не будем. При этом, как уже было отмечено, мы ниже в разделе IV (см. также рис.1) рассмотрим определение применяемого в настоящей статье варианта лепестковых автоматов – необходимых для одной из формулировок гипотезы ($\mathfrak{A}$).

Сначала рассмотрим приведённые в предыдущих статьях определения, связанные с бинарным отношением $\lessdot\!\!\!>$. Если для конечных языков A и B выполнено условие

$$(\forall u \in A^*)\ (\exists v \in B^*)\ (u \in \mathrm{opref}(v)),$$

будем писать $A \lessdot B$ (либо $B \gtrdot A$). Если одновременно выполнены условия $A \lessdot B$ и $A \gtrdot B$, будем писать $A \lessdot\!\!\!> B$. Наоборот, в случае, когда уже известно, что условие $A \gtrdot B$ не выполнено, мы в случае выполнения условия $A \lessdot B$ можем также писать $A \lessdot B$ (либо $B \rhd A$) [2].

Далее рассмотрим обозачения, связанные с максимальными префиксными кодами и их морфизмами. Для рассматриваемого алфавита Δ множество максимальных префиксных кодов над Δ [12, стр. 135, 144] будем обозначать $\mathfrak{mp}(\Delta)$. [3]

Множество конечных языков над алфавитом Δ, каждый из которых в качестве подмножества (возможно, несобственного) содержит некоторый максимальный префиксный код над Δ, будем обозначать $\mathfrak{mp}^+(\Delta)$ – т. е. формально

$$\mathfrak{mp}^+(\Delta) = \left\{ A_\Delta \subseteq \Sigma^* \;\middle|\; (\exists B_\Delta \subseteq A_\Delta)\,(B_\Delta \in \mathfrak{mp}(\Delta)) \right\}.$$

Морфизм (использование этого термина согласовано с [13]) – это отображение

$$h : \Delta^* \to \Sigma^*,$$

для которого:

- для каждой буквы $d \in \Delta$ её образ $h(d) \in \Sigma^*$ задаётся;
- а для каждого слова $d_1 d_2 \dots d_n \in \Delta^*$ полагаем

$$h(d_1 d_2 \dots d_n) = h(d_1)\, h(d_2) \dots h(d_n).$$

Нам будут очень важны морфизмы, соответствующие конечным языкам. То есть для некоторого языка

$$A \in \Sigma^*, \quad A = \{u_1, u_2, \dots, u_n\}$$

[2] Также можно очевидным образом объяснить обозначение $A \lessdot\!\!> B$ (соответствующее ему условие выполнено, например, для пары $A = \{a\}$, $B = \{b\}$) – однако вряд ли это обозначение будет когда-либо востребовано.

[3] Возможный несложный *недетерминированный* алгоритм построения *любого* такого максимального префиксного кода – т. е. любого элемента множества $\mathfrak{mp}(\Delta)$ – следующий.

- (База.) Максимальным префиксным кодом является сам язык Δ. (Заметим, что в некоторых случаях удобнее в качестве базы рассматривать язык $\{\varepsilon\}$ – но в нашей ситуации этого делать нельзя.)
- (Шаг.) Если известно, что $A_\Delta \in \mathfrak{mp}(\Delta)$, и при этом $u_\Delta \in A_\Delta$, то считаем, что $(A_\Delta \setminus \{u_\Delta\}) \cup \{u_\Delta\}\Delta \in \mathfrak{mp}(\Delta)$.

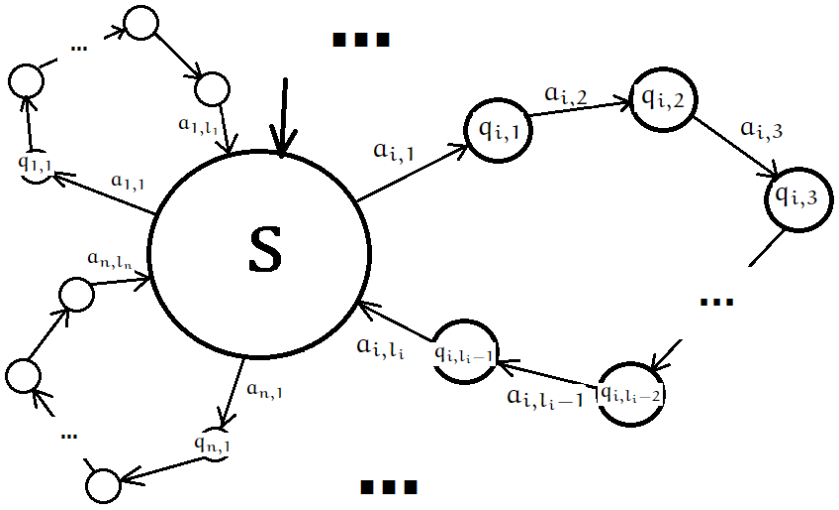

Рис. 1. Подробная общая схема графа переходов лепесткового автомата. В варианте, рассматриваемом в настоящей статье, все состояния (включая «главное», s) являются выходными.

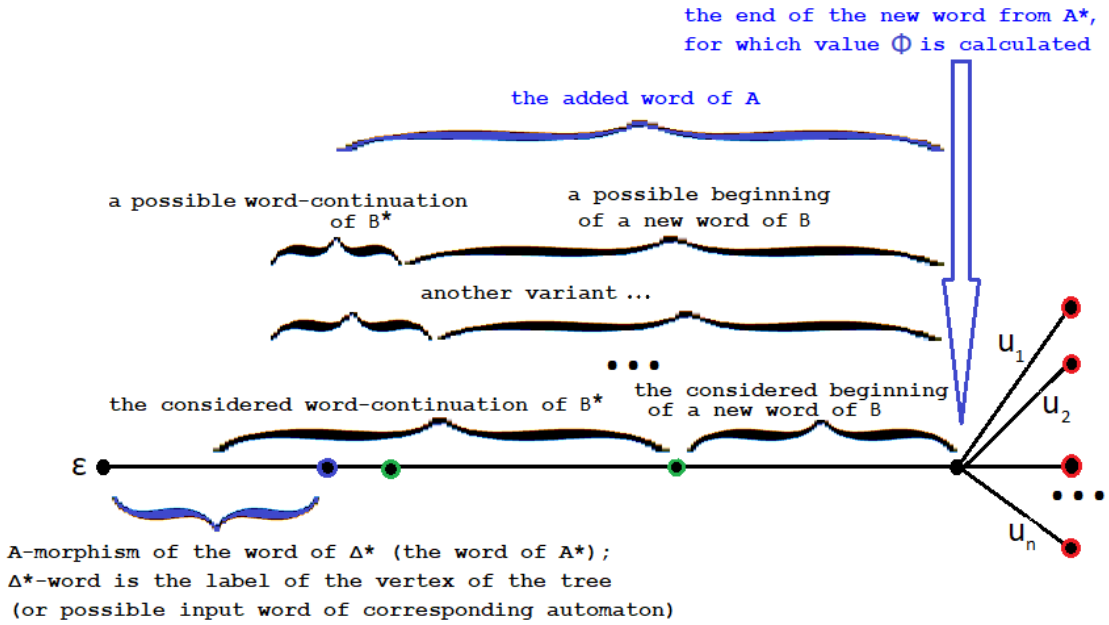

Рис. 2. Первая часть описания алгоритма построения функции Φ: все слова-продолжения из A^*.

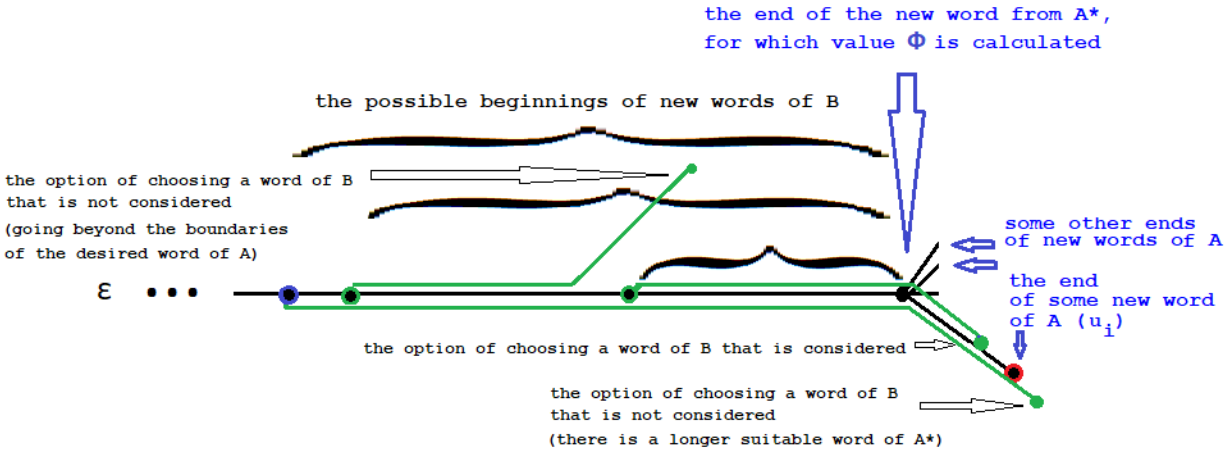

Рис. 3. Вторая часть описания алгоритма построения функции Φ: выбор некоторого слова-продолжения из A^* и соответствующего суффикса нового слова языка B (последнее эквивалентно «префиксу нового слова языка A»).

мы рассматриваем алфавит

$$\Delta_A = \{ d_1, d_2, \ldots, d_n \}$$

(если это не вызывает неоднозначностей, пишем обычно просто Δ), а для него – морфизм

$$h_A : \Delta_A^* \to \Sigma^*,$$

задающийся следующим образом[4]:

$$h_A(d_1) = u_1\,,\ h_A(d_2) = u_2\,,\ \ldots,\ h_A(d_n) = u_n\,.$$

Если A не обладает свойством префикса[5], то определённый здесь морфизм также будем называть *непрефиксным*.

Для рассматриваемых алфавита Δ и языка $A \subseteq \Sigma^*$ следующее множество языков над алфавитом Σ будем обозначать $\mathrm{mp}(A)$:

$$\mathrm{mp}(A) = \big\{\, B \in \Sigma^* \,\big|\, (\exists A_\Delta \in \mathrm{mp}(\Delta))\,(B = h_A(A_\Delta)) \,\big\}.$$

Аналогично,

$$\mathrm{mp}^+(A) = \big\{\, B \in \Sigma^* \,\big|\, (\exists A_\Delta \in \mathrm{mp}^+(\Delta))\,(B = h_A(A_\Delta)) \,\big\}.$$

B. Предыдущие версии гипотезы (Ξ)

Теперь приведём все эквивалентные «старые версии» гипотезы (Ξ), [8] и др.

Гипотеза Ξ . Для любых конечных языков $A, B \subseteq \Sigma^*$ выполнено следующее. Эквивалентность $A \lessdot\!\!\gtrdot B$ выполняется тогда и только тогда, когда существует язык $D \subseteq \Sigma^*$, такой что при рассмотрении соответствующего алфавита Δ (где $|\Delta| = |D|$) существуют расширенные максимальные префиксные коды

$$A_\Delta, B_\Delta \subseteq \Delta^*,$$

такие что

$$A = h_D(A_\Delta)\,,\quad B = h_D(B_\Delta). \qquad \square$$

Гипотеза Ξ′. Пусть задан некоторый конечный язык $D \subseteq \Sigma^*$. При этом D – минимальный в своём классе эквивалентности; это означает, что не существует языка $\mathcal{D} \subseteq \Sigma^*$, такого что $D \in \mathrm{mp}^+(\mathcal{D})$.

Любой конечный язык $A \subseteq \Sigma^*$, принадлежащий рассматриваемому классу эквивалентности $A \lessdot\!\!\gtrdot D$, может быть построен за следующие 4 шага:

1) выбирается некоторый алфавит Δ, такой что $|\Delta| = |D|$;
2) строится некоторый язык $A'_\Delta \in \mathrm{mp}(\Delta)$;
3) строится некоторый (любой – но, конечно, зависящий от исходного языка A) язык $A_\Delta \subseteq \Delta^*$, такой что $A_\Delta \supseteq A'_\Delta$; т. е., неформально, к языку A'_Δ добавляются произвольные слова, а формально – $A_\Delta \in \mathrm{mp}^+(\Delta)$;
4) строится язык $h_D(A_\Delta)$.

Построенный язык должен совпасть с языком A. $\square$

Гипотеза Ξ″. Утверждение, сформулированной в [5, Th. 1], является верным – т. е. сформулированное в [5] достаточное условие эквивалентности $A \lessdot\!\!\gtrdot B$ является необходимым и достаточным. $\square$

Гипотеза Ξ‴. Не существует пары (D, u), где

$$D \subseteq \Sigma^* \quad \text{и} \quad u \in \Sigma^*,\ u \notin D^*,$$

такой что:

- язык D – минимальный в своём классе эквивалентности;
- $(D \cup \{u\}) \lessdot\!\!\gtrdot D$. $\square$

C. Конкретная версия процедуры детерминизации автомата

В заключение раздела приведём описание конкретного варианта процедуры детерминизации: такая конкретизация нужна потому, что впоследствии (в разделе VI) мы воспользуемся каноническими автоматами для очередной эквивалентной формулировки гипотезы (Ξ). Начиная с 1950-х подобная процедура в литературе описывалась много раз, поэтому конкретные ссылки вряд ли представляют интерес; однако, мы будем использовать автоматы, к которым эта процедура уже была применена, и поэтому желательно ориентироваться на конкретную её версию. При этом «вторую часть» алгоритма (т. е. «канонизацию» автомата, иными словами – конкретный вариант процедуры объединения эквивалентных состояний) мы опустим: она не представляет интереса для материала настоящей статьи. Некоторые подробности можно посмотреть в [14], [15], [16], [17].

Для заданного недетерминированного конечного автомата

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

мы последовательно формируем эквивалентный ему детерминированный автомат

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

следующим образом. Во-первых, полагаем

$$\hat{Q},\, \hat{F} \subseteq \mathcal{P}(Q).$$

(здесь и далее мы используем обозначения (1) – в связи с этим обозначение стартового состояния нового автомата $\{S\}$ корректно).

Далее, для *базиса* индукции такого формирования мы полагаем

$$\hat{\delta} = \varnothing, \quad \hat{Q} = \{S\};$$

эти множества, вообще говоря, будут изменяться. Теперь опишем *шаг* индукционного формирования нового автомата.

Если мы уже построили значения $\hat{\delta}$ для любого состояния множества $\hat{Q}$ и для всех букв алфавита Σ, то мы определяем множество

$$\hat{F} = \big\{\, \hat{q} \in \hat{Q} \,\big|\, (\exists f \in \hat{q})\,(f \in F) \,\big\}$$

и завершаем процесс построения автомата $\hat{K}$.

В противном случае мы выбираем некоторое состояние $\hat{q} \in \hat{Q}$, для которого ещё не сформирован переход $\hat{\delta}(\hat{q}, a)$; отметим, что это мы делаем для каждой буквы $a \in \Sigma$. А именно,

$$\hat{\delta}(\hat{q}, a) = \bigcup_{q \in \hat{q}} \delta(q, a). \qquad (2)$$

[4] Мы почти всегда рассматриваем язык A как множество – а *не* как упорядченное множество. Однако неточностей во вводимых обозначениях нет, поскольку мы в приведённой выше формуле фактически «перенумеровали» слова языка A.

[5] Т. е. $(\exists u, v \in A)\,(u \in \mathrm{opref}(v))$.

К описанным действиям приведём два дополнительных комментария:

- в (2) мы использовали знак $=$, а не «обычный» для подобных ситуаций знак $\ni$, – поскольку после выполнения (2) мы ничего иного в множество $\hat{\delta}(\hat{q}, a)$ добавлять не будем;
- как мы уже отмечали, обычные дальнейшие эквивалентные преобразования автомата – т. е. процесс объединения эквивалентных состояний и получения таким образом автомата канонического – для настоящей статьи интереса не представляют; однако в дальнейших примерах мы всё-таки будем рассматривать именно канонические автоматы.

III. ЕЩЁ РАЗ О ПОСТРОЕНИИ ФУНКЦИИ ПЕРЕХОДОВ АВТОМАТА PRI

Построение функции Φ мы вынесли в отдельный раздел – несмотря на то, что здесь, кроме её определения, приведены только два рисунка и комментарии к ним, а похожие рисунки нами рассматривались и ранее, например в [1]. Однако чисто формально Φ можно рассматривать не только как выбор дальнейшего множества слов для анализа возможного выполнения отношений $\trianglelefteq$ (что было сделано в [10], где, однако, применялись иные обозначения, а также впоследствии в [1], [2]) – но и как функцию переходов детерминированного конечного автомата PRI; именно на последний вариант мы и ориентируемся в настоящем разделе. Таким образом можно сказать, что функция переходов автомата PRI очень связана с темой статьи – и особенно она связана с поиском требуемого множества состояний в недетерминированном лепестковом автомате, раздел IV.

Итак, сначала повторим определение этой функции из [3] – именно как функции переходов детерминированного конечного автомата.

Для пары непустых конечных языков $A, B \subseteq \Sigma^*$, таких что $A \not\ni \varepsilon$ и $B \not\ni \varepsilon$, язык

$$\underset{A-B}{\Phi}(u) \subseteq \Sigma^*$$

строится для некоторого слова $u \in \Sigma^*$ по следующему алгоритму – в нём используется вспомогательный язык D' и формируется язык-ответ D.

1) Для начала работы алгоритма полагаем
$$D' = \{u\}A, \quad D = \varnothing.$$
2) Если $D' = \varnothing$, то выход из алгоритма с ответом D.
3) Выбираем некоторое слово $v \in D'$, исключая его из D'.
4) Для выбранного v рассматриваем все варианты его представления в виде $v = uv'$, где $u \in B^+$,
$$D \mathrel{+}: v'$$
(т. е. включаем каждое такое v' в язык D).
5) Если
$$(\exists w \in B)\,(v \in \operatorname{opref}(w)),$$
то $D \mathrel{+}: v$ (т. е. включаем v в язык D)[6].
6) Переходим на п. 2.

[6] Иначе про v «забываем».

(Отметим, что согласно приведённому алгоритму про итоговый «выходной» язык $D = \underset{A-B}{\Phi}(u)$ можно утверждать, что $D \subseteq \operatorname{opref}(B)$.)

Приведённое определение фактически формулирует алгоритм построения функции F из [1] – мы также называли значения этой функции «множествами хвостов». Заметим также, что на шаге 4 мы могли бы заменить условие $u \in B^+$ на $u \in B$ – но в этом случае нам бы пришлось иногда рассматривать «лишние» слова[7].

Для языка $C \subseteq \Sigma^*$ определяем

$$\underset{A-B}{\Phi}(C) = \bigcup_{u \in C} \underset{A-B}{\Phi}(u).$$

(Аналогично предыдущему, про итоговый язык можно утверждать, что $\underset{A-B}{\Phi}(C) \subseteq \operatorname{opref}(B)$.)

Далее будем рассматривать приведённые выше рисунки 2 и 3 – которые можно назвать описанием алгоритма построения функции Φ:

- «общую часть» этого алгоритма (рис. 2),
- и выбор некоторого слова-продолжения (рис. 3) – «особенную часть».

Те комментарии, которые «поместились» на рисунках, приведены на английском языке – мы надеемся, что это не представит проблем.

Начнём с «общей части» алгоритма формирования Φ, рис. 2. Синим отмечены слова языка A^* (ещё точнее – значения h_A-морфизма слов языка над алфавитом Δ); в частности, единственная синяя точка – это «главное» слово, т. е. слово, для которого вычисляется значение функции Φ (в обозначениях алгоритма – слово u). К нему добавляется справа некоторое слово, тоже из языка A («лежащая» синяя фигурная скобка, её пометка – “the added word of A”; иными словами, это одно из слов – представителей языка A на шаге 1 алгоритма вычисления Φ); конец этого слова отмечен направленной вниз большой синей стрелкой.

Однако *до* этого добавления-конкатенации «слову синей точки» u соответствует «множество хвостов»: это префиксы (до этой синей точки) слов-продолжений из B^* (надпись “a possible word-continuation of B*”). Из них мы вибираем одно (но, конечно, в процессе работы алгоритма должны рассмотреть их все) – и это выбранное слово отмечено на рисунке как “the considered word-continuation of B*”.

К нему нужно слово-продолжение, также из B, – и начало этого слово помечено на рисунке как “the considered beginning of a new word of B”. А все возможные продолжения этого слова (составляющие вместе с ним, т. е. приписанные к нему справа, слова из B) – это и есть значения функции переходов Φ; они на рисунке обозначены красными точками и словами $u_1, u_2, \ldots, u_n$.

Теперь перейдём к «особенной части» алгоритма формирования Φ, рис. 3. Он, в отличие от первого рисунка, почти не изменился (по сравнению с рисунком, приведённым в [1]). Но мы всё-таки приводим его, причём в немного обновлённом виде, – также с английскими надписями и существенно более подробными комментариями. При этом комментарии ориентированы (как

[7] Это повлияло бы на описание конечных автоматов и соответствующих им алгоритмов, использующих язык $\underset{A-B}{\Phi}(u)$: они стали бы «более громоздкими», но их содержательный смысл не изменился бы.

и для первого рисунка) также на построение функции переходов конечного автомата PRI.

Этот второй рисунок (рис. 3) можно считать частным случаем первого. При этом не только к «слову синей точки» u, но и к другим возможным словам из B^*, имеющим u в качестве префикса, мы пытаемся добавить слова-продолжения (отмеченные на рисунке зелёными ломаными линиями).

Такие попытки в двух случаях (в «верхнем» и «нижнем») – «не удаются», оба раза мы пишем “the option of choosing a word of B that is not considered”. При этом в «верхнем» случае происходит «слишком ранний» выход за границы слова, отмеченного направленной вниз большой синей стрелкой (“going beyond the boundaries of the desired word of A”), а в «нижнем» случае – «слишком поздний» (“there is a longer suitable word of A$*$”). Ещё точнее, в «нижнем» случае у нас образуется другое слово-кандидат (отмеченное красной точкой) – которое и надо рассматривать вместо u.

И только «средний» случай нам подходит (“the option of choosing a word of B that is considered”) – он отмечен зелёной точкой, не имеющей внутри себя чёрной. Это и есть одно из слов, названных u_i на рис. 2; точнее, сам выбор подслова, которое мы считаем одним из u_i, легко понять на основе рис. 3.

В заключение раздела ещё раз отметим, что рассмотренная здесь функция Φ в наших предыдущих публикациях использовалась:

- и как вспомогательная функция проверки условия $A \trianglelefteqslant B$ (а именно – как функция построения вспомогательных множеств слов для этого алгоритма);
- и как функция переходов конечного автомата PRI – что верно и для настоящей статьи.

IV. ПРИМЕНЕНИЕ ЛЕПЕСТКОВЫХ АВТОМАТОВ ДЛЯ ФОРМУЛИРОВКИ ГИПОТЕЗЫ ($\mathfrak{A}$)

Рассмотрим лепестковые автоматы немного подробнее. Сразу должен возникнуть вопрос: какое они отношение могут иметь к рассматриваемому нами *комплексу задач*? Ведь основная цель практически всех задач, описанных в [9], – исследовать бинарное отношение $\trianglelefteqslant$, причём в предыдущих публикациях мы отмечали, что возможны два варианта такого исследования:

- либо получить некоторые новые необходимые и / или достаточные условия выполнения этого отношения;
- либо описать некоторые новые возможности для его применения в других задачах теории формальных языков.

Но ведь в обоих случаях необходимыми «входными данными» являются *два* конечных языка (а не один) …

Однако ответ на поставленный вопрос тоже почти очевиден: да, для всех описанных в [9] задач мы действительно рассматриваем пары конечных языков; при этом для каждого из языков такой пары желательно (и обычно возможно) заменить его рассмотрение на рассмотрение «минимального» языка – «минимального» в соответствующем ему классе эквивалентности по отношению $\trianglelefteqslant$. Отметим, что возможные подходы к определению понятия «минимальность» обсуждались, например, в [7].

Перейдём к непосредственному определению того варианта лепестковых автоматов, который будет использован в настоящей статье[8]. Путь задан некоторый конечный язык

$$\begin{gathered} A = \{u_1, u_2, \ldots, u_i, \ldots, u_n\}, \\ \text{где } u_1, \ldots, u_i, \ldots, u_n \in \Sigma^*, \end{gathered} \tag{3}$$

причём $A \not\ni \varepsilon$. Пусть также

$$\begin{gathered} u_i = a_{i,1} a_{i,2} \ldots a_{i,l_i} \quad (l_i \geqslant 1) \\ \text{для каждого } i = 1, 2, \ldots, n. \end{gathered}$$

Для этого языка A рассмотрим недетерминированный конечный автомат

$$\mathcal{K}(A) = (Q, \Sigma, \delta, \{s\}, Q), \tag{4}$$

где

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

а функция переходов δ для каждого $i = 1, \ldots, n$ определена описанным далее способом.

- Если $|u_i| = 1$, то $\delta(s, a_{i,1}) \ni s$.
- Иначе[9]:
  - $\delta(s, a_{i,1}) \ni q_{i,1}$;
  - $\delta(q_{i,l_i-1}, a_{i,l_i}) = \{s\}$;
  - для каждого j, такого что $2 \leqslant j \leqslant l_i - 1$, полагаем $\delta(q_{i,j-1}, a_{i,j}) = \{q_{i,j}\}$.

Такая запись в точности описывает автомат, приведённый на рис. 1; также очевидно, что автомат (4) определяет язык $\mathrm{pref}(A^*)$.

Как уже было отмечено во введении, предмет настоящей статьи, по-видимому, довольно простой: как можно несложным образом сформулировать проверку выполнения гипотезы *только для одного (заданного конкретного) конечного языка* A. Для этого мы и строим автомат $\mathcal{K}(A)$.

В качестве примера рассмотрим язык, который в примерах предыдущих публикаций рассматривался «в качестве второго», т. е. в наших обычных обозначениях – как язык B; а именно,

$$B = \{aaaa, abb, abba, bbb\}. \tag{5}$$

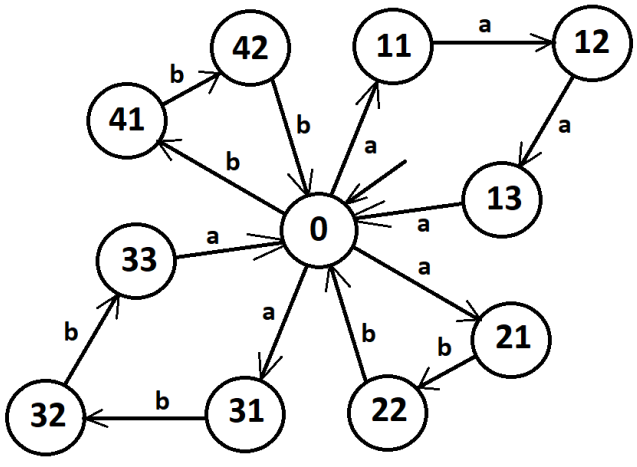

Рис. 4. Лепестковый автомат для рассматриваемого языка B.

[8] Напомним, что в наших предыдущих публикаций было использовано несколько похожих определений.

[9] Мы считаем, что $A \not\ni \varepsilon$, – поэтому $|u_i| \geqslant 1$.

Мы получаем автомат, приведённый на рис. 4; аналогично рис. 1, все его состояния являются выходными. Как видно из этого рисунка, мы для упрощения обозначений состояние s обозначили как 0, а вместо каждого $q_{i,j}$ пишем ij (здесь это возможно, т. к. максимальное значение номера состояния равно 4). Мы продолжим рассмотрение этого автомата в оставшейся части статьи.

V. ФОРМУЛИРОВКА ВАРИАНТА ГИПОТЕЗЫ ($\mathfrak{A}$) ДЛЯ НЕДЕТЕРМИНИРОВАННОГО АВТОМАТА

В этом разделе приводится формулировка варианта гипотезы ($\mathfrak{A}$), использующая недетерминированный автомат $\mathcal{K}(A)$.

Заранее отметим, что мы не будем доказывать эквивалентность приведённой ниже формулировки ($\mathfrak{A}^{IV}$) и формулировок, приведённых ранее. Возможная схема этого доказательства такова. При отрицании формулировки ($\mathfrak{A}'''$) мы рассматриваем лепестковый автомат $\mathcal{K}(D)$ для использующегося в этой формулировке языка D, а слово u имеет тот же смысл, что и ниже. Это слово является входным для, вообще говоря, нескольких состояний автомата $\mathcal{K}(D)$ – и при этом важно, чтобы среди этих состояний не было «главного» состояния лепесткового автомата (в противном случае мы имеем слово из D^*). Однако объединение выходных языков этих состояний содержит язык исходного лепесткового автомата как подмножество – что и формулируется в этой версии гипотезы.

Гипотеза $\mathfrak{A}^{IV}$. *Не существует* конечного языка $A \subseteq \Sigma^*$, такого что выполнено следующее утверждение.

Для автомата $\mathcal{K}(A)$, его языка $L = \mathcal{L}(\mathcal{K}(A))$ и его множества состояний Q существует слово

$$u \in L \setminus A^*,$$

такое что для подмножества множества состояний

$$Q' = \left\{ q \in Q \,\middle|\, \mathcal{L}^{in}_{\mathcal{K}(A)}(q) \ni u \right\}$$

выполнено такое условие:

$$\bigcup_{q \in Q'} \mathcal{L}^{out}_{\mathcal{K}(A)}(q) \supseteq A^*. \qquad \square$$

Особо отметим, что такая проверка (т.е. проверка возможного существования требуемого слова u) возможна *для конкретного автомата* $\mathcal{K}(A)$, или, иными словами, *для конкретного языка* A. Кроме того, очевидно, что у авторов нет примеров таких языков (иначе нам не нужно было бы формализовать всё это в качестве гипотезы). Таким образом, для всех рассмотренных на данный момент языков условие, сформулированное в гипотезе, не выполняется.

Рассмотрим пример (частный случай) – для лепесткового автомата, приведённого на рис. 4, и слова $abbab$. Соответствующие ему *входные* состояния – 22, 32 и 41 (отметим, что состояния 0 среди них нет, это нам подходит), поэтому для проверки *частного случая* гипотезы надо показать, что объединение *выходных* языков этих состояний не содержит языка исходного лепесткового автомата в качестве подмножества. А последнее следует хотя бы из того, что все слова выходных языков этих состояний начинаются с b – в то время как в языке исходного автомата присутствуют и слова, начинающиеся с a.

Можно также отметить, что все различные варианты множеств вершин Q' (соответствующие различным входным словам u) могут быть получены в процессе детерминизации рассматриваемого автомата $\mathcal{K}(A)$, причём при использовании обычных алгоритмов; напомним, что один из таких алгоритмов приведён в разделе II. (При применении алгоритма детерминизации, в котором рассматриваются все подмножества состояний заданного автомата $\mathcal{K}(A)$ в качестве так называемых агрегатных состояний, нам необходимо учитывать только те агрегатные состояния, которые являются достижимыми.) Всё это позволяет сформулировать другую эквивалентную версию рассматриваемой нами гипотезы; её мы рассматриваем в следующем разделе – формулируя тем самым ещё одну эквивалентную версию гипотезы ($\mathfrak{A}$).

VI. ФОРМУЛИРОВКА ВАРИАНТА ГИПОТЕЗЫ ($\mathfrak{A}$) ДЛЯ КАНОНИЧЕСКОГО АВТОМАТА

Гипотезу, про которую сказано в конце предыдущего раздела, мы будем формулировать для канонических автоматов (а не для таких детерминированных, которые получаются путём детерминизации – но без «канонизации» – автоматов лепестковых); по-видимому, разница непринципиальна, поскольку само объединение эквивалентных состояний детерминированного автомата для настоящей статьи интереса не представляет.

Гипотеза $\mathfrak{A}^{V}$. *Не существует* конечного языка $A \subseteq \Sigma^*$, такого что выполнено следующее утверждение.

Для автомата $\mathcal{K}(A)$ строится эквивалентный канонический автомат $\hat{\mathcal{K}}(A)$ с использованием стандартной процедуры детерминизации (раздел II); мы будем использовать некоторые обозначения, приведённые в описании этой процедуры.

В автомате $\hat{\mathcal{K}}(A)$ мы выбираем некоторое состояние $\hat{q} \in \hat{Q}$, такое что $\hat{q} \not\ni s$, а для него рассматриваем детерминированный автомат [10]

$$\hat{\mathcal{K}}_{\hat{q}}(A) = (\, \hat{Q},\, \Sigma,\, \hat{\delta},\, \{\hat{q}\},\, \hat{F}\,).$$

Для последнего автомата приведём такие два вспомогательных комментария:

- напомним, что состояние s, использовавшееся для выбора $\hat{q}$, является «главным» состоянием лепесткового автомата $\mathcal{K}(A)$;
- при таких ограничениях по крайней мере одно такое состояние $\hat{q}$ даёт выполнение рассматриваемого условия (*не* выполнение которого и постулируется в настоящей версии гипотезы).

В этом случае

$$\mathcal{L}(\hat{\mathcal{K}}_{\hat{q}}(A)) \;\; \supseteq \;\; \mathcal{L}(\hat{\mathcal{K}}(A)) = \mathcal{L}(\mathcal{K}(A)) = A^*. \qquad \square$$

Продолжим рассмотрение примера автомата, приведённого на рис. 4. Сначала рассмотрим процесс детерминизации этого автомата – см. таблицу I. Сразу отметим, что в ней мы обозначали агрегатные состояния, применяя такие соглашения:

[10] Можно показать, что он будет *для своего языка* каноническим – раз таковым был исходный для него автомат $\hat{\mathcal{K}}(A)$.

Таблица I
Процесс детерминизации исходного лепесткового автомата.

		a	b
→	0	11 21 31	41
	11 21 31	12	22 32
	41	—	42
	12	13	—
	22 32	—	0 33
	42	—	0
	13	0	—
	0 33	0 11 21 31	41
	0 11 21 31	11 12 21 31	22 32 41
	11 12 21 31	12 13	22 32
	22 32 41	—	0 33 42
	12 13	0 13	—
	0 33 42	0 11 21 31	0 41
	0 13	0 11 21 31	41
	0 41	11 21 31	41 42
	41 42	—	0 42
	0 42	11 21 31	0 41

- жирным шрифтом обозначены те из них, которые не содержат состояния 0 в качестве своего элемента;
- из последних – серым фоном выделены те, в которых имеются переходы по обеим буквам алфавита (из отсутствия хотя бы одного такого перехода сразу следует, что автомат с таким стартовым состоянием заведомо не подходит, т. е. не требует дальнейшей проверки); таковых состояний 2.

Таблица II
Полученный эквивалентный детерминированный автомат.

	K	a	b
→	A	B	C
	B	D	E
	C	—	F
	D	G	—
	E	—	H
	F	—	A
	G	A	—
	H	J	C
	J	K	L
	K	M	E
	L	—	N
	M	P	—
	N	J	Q
	P	J	C
	Q	B	R
	R	—	S
	S	B	Q

Также напомним, что согласно нашим договорённостям все состояния рассматриваемых автоматов являются выходными – поэтому в таблицах мы этот факт тоже не отмечаем.

Обычным образом переобозначая агрегатные состояния, получаем эквивалентный детерминированный автомат, приведённый в таблице II [11].

Мы уже знаем, что для проверки частного случая гипотезы нужно рассмотреть 2 автомата – получающиеся из таблицы II путём замены стартового состояния на одно из тех двух, которые мы ранее отметили серым фоном; в новых обозначениях это состояния B и K.

Для входного состояния B мы получаем такой автомат (K_1, таблица III):

Таблица III
Первый проверяемый автомат.

	K1	a	b	
	A	B	C	A⩽B ⇒
→	B	D	E	B⩽D & C⩽E ⇒
	C	—	F	... E⩽Ø ...
	D	G	—	
	E	—	H	
	F	—	A	
	G	A	—	
	H	J	C	
	J	K	L	
	K	M	E	
	L	—	N	
	M	P	—	
	N	J	Q	
	P	J	C	
	Q	B	R	
	R	—	S	
	S	B	Q	

Справа от таблицы приведено краткое объяснение того, что этот автомат не является контрпримером к гипотезе $\mathbb{A}^V$; в этом тексте знак ⩽ означает, что выходной язык «меньшего» состояния *должен быть* подмножеством языка состояния «большего». Должно быть выполнено условие $\mathcal{L}^{out}_{K_1}(A) \subseteq \mathcal{L}^{out}_{K_1}(B)$, а отсюда, рассматривая оба возможных перехода из этих состояний, получаем необходимость выполнения условия

$$\mathcal{L}^{out}_{K_1}(B) \subseteq \mathcal{L}^{out}_{K_1}(D) \;\;\&\;\; \mathcal{L}^{out}_{K_1}(C) \subseteq \mathcal{L}^{out}_{K_1}(E);$$

первое из условий в этой конъюнкции невозможно из-за требуемых переходов по букве b и заведомого невыполнения условия $\mathcal{L}^{out}_{K_1}(E) = \varnothing$.

Второй необходимый для проверки гипотезы автомат (K_2, его вход – состояние K) приведён в таблице IV; проверка осуществляется аналогично.

[11] Как мы уже отмечали, он является каноническим. Совпадение имени автомата с именем его состояния сделано для удобства чтения – и не должно вызвать проблем.

Таблица IV
Второй проверяемый автомат.

	K2	a	b	
	A	B	C	A⩽K ⇒
	B	D	E	B⩽M & C⩽E ⇒
	C	—	F	... E⩽Ø ...
	D	G	—	
	E	—	H	
	F	—	A	
	G	A	—	
	H	J	C	
	J	K	L	
→	K	M	E	
	L	—	N	
	M	P	—	
	N	J	Q	
	P	J	C	
	Q	B	R	
	R	—	S	
	S	B	Q	

VII. ЗАКЛЮЧЕНИЕ

Возможные направления дальнейших работ по этой тематике опишем очень кратко; можно также сказать, что все они связаны с проблемами, приведёнными во введении.

Как мы уже отмечали, бессмысленно ставить вопрос о возможном существовании *полиномиального* алгоритма проверки частного случая гипотезы (Ѫ): даже если мы не строим эквивалентный канонический автомат (аналогичный либо эквивалентный автомату NSPRI), то мы всё равно делаем аналогичную работу (например – строим автомат, аналогичный автомату PRI, имеющий, по сравнению с автоматом NSPRI, «экспоненициально много» состояний). Кроме того, за полиномиальное время мы, вообще говоря, не сможем проверить совпадение языка автомата NSPRI с Σ^*.

Список литературы

[1] Мельников Б., Мельникова А. *Бесконечные деревья в алгоритме проверки условия эквивалентности итераций конечных языков. Часть I* // International Journal of Open Information Technologies. – 2021. – Vol. 9. No. 4. – P. 1–11.

[2] Мельников Б., Мельникова А. *Бесконечные деревья в алгоритме проверки условия эквивалентности итераций конечных языков. Часть II* // International Journal of Open Information Technologies. – 2021. – Vol. 9. No. 5. – P. 1–11.

[3] Мельников Б. *Варианты конечных автоматов, соответствующих бесконечным итерационным деревьям морфизмов. Часть I* // International Journal of Open Information Technologies. – 2021. – Vol. 9. No. 7. – P. 5–13.

[4] Мельников Б. *Варианты конечных автоматов, соответствующих бесконечным итерационным деревьям морфизмов. Часть II* // International Journal of Open Information Technologies. – 2021. – Vol. 9. No. 10. – P. 1–8.

[5] Melnikov B. *The equality condition for infinite catenations of two sets of finite words* // International Journal of Foundation of Computer Science. – 1993. – Vol. 4. No. 3. – P. 267–274.

[6] Мельников Б. *Полурешётки подмножеств потенциальных корней в задачах теории формальных языков. Часть I. Извлечение корня из языка* // International Journal of Open Information Technologies. – 2022. – Vol. 10. No. 4. – P. 1–9.

[7] Мельников Б. *Полурешётки подмножеств потенциальных корней в задачах теории формальных языков. Часть II. Построение инверсного морфизма* // International Journal of Open Information Technologies. – 2022. – Vol. 10. No. 5. – P. 1–8.

[8] Мельников Б. *Полурешётки подмножеств потенциальных корней в задачах теории формальных языков. Часть III. Условие существования решётки* // International Journal of Open Information Technologies. – 2022. – Vol. 10. No. 7. – P. 1–9.

[9] Мельников Б. *О комплексе задач исследования необходимых условий равенства бесконечных итераций конечных языков* // International Journal of Open Information Technologies. – 2023. – Vol. 11. No. 1. – P. 1–12.

[10] Мельников Б. *Алгоритм проверки равенства бесконечных итераций конечных языков* // Вестник Московского университета, серия 15 («Вычислительная математика и кибернетика»). – 1996. – № 4. – С. 49–54.

[11] Melnikov B. *Some more on ω-finite automata and ω-regular languages. Part I: The main definitions and properties* // International Journal of Open Information Technologies. – 2020. – Vol. 8. No. 8. – P. 1–7.

[12] Лаллеман Ж. *Полугруппы и комбинаторные приложения*. – М., Мир. – 1985. – 440 с.

[13] Саломаа А. *Жемчужины теории формальных языков*. – М., Мир. – 1986. – 159 с.

[14] Мельников Б. *Подклассы класса контекстно-свободных языков (монография)*. – М., Изд-во Московского университета. – 1995. – 174 с. – ISBN 5-211-03448-1.

[15] Мельников Б. *Регулярные языки и недетерминированные конечные автоматы (монография)*. – М., Изд-во Российского государственного социального университета. – 2018. – 179 с. – ISBN 978-5-7139-1355-7.

[16] Мельников Б., Долгов В. *Упрощённые регулярные языки и специальное отношение эквивалентности на классе регулярных языков. Часть I* // International Journal of Open Information Technologies. – 2022. – Vol. 10. No. 9. – P. 12–20.

[17] Мельников Б., Долгов В. *Упрощённые регулярные языки и специальное отношение эквивалентности на классе регулярных языков. Часть II* // International Journal of Open Information Technologies. – 2023. – Vol. 11. No. 1. – P. 13–19.

Борис Феликсович МЕЛЬНИКОВ,
профессор Совместного университета МГУ – ППИ в Шэньчжэне, Китай
(http://szmsubit.ru/),
email$_1$: bormel@smbu.edu.cn,
email$_2$: bf-melnikov@yandex.ru,
mathnet.ru: personid=27967,
elibrary.ru: authorid=15715,
scopus.com: authorId=55954040300,
ORCID: orcidID=0000-0002-6765-6800.

Александра Александровна МЕЛЬНИКОВА,
доцент Димитровградского инженерно-технологического института – филиала Национального исследовательского ядерного университета «МИФИ»
(https://diti-mephi.ru/),
email: super-avahi@yandex.ru,
mathnet.ru: personid=148963,
elibrary.ru: authorid=143351,
scopus.com: authorId=6603567251,
ORCID: orcidID=0000-0002-1658-6857.

The use of petal finite automata to verify the fulfillment of a special case of the Zyu hypothesis (for a given finite language)

Boris Melnikov, Aleksandra Melnikova

***Abstract*—In this paper, we continue to study the properties of a special binary equivalence relation at infinity. Before, we described one hypothesis of the theory of formal languages, which we called the Zyu hypothesis; one of some its equivalent formulations can be expressed as follows. For two finite languages A and B without empty words, we can write the necessary and sufficient condition that the iteration of any of these languages belongs to the set of prefixes of the second language as follows: there is some alphabet (different from the alphabet over which A and B are given), over it there are two certain maximal prefix codes (generally speaking, different ones) and two languages containing these maximal prefix codes as subsets, as well as some morphism between the two considered alphabets. Then, applying this morphism to the last two languages, we obtain the given languages A and B.**

We consider an equivalent formulation of this hypothesis using not two languages over the given alphabet, but only one; at the same time, it is important to note that such a variant of the Zyu hypothesis consists in fulfilling some condition for any finite language. Using the above formulation, the subject of this article can be briefly described as follows: how can a similar version of the Zyu hypothesis be tested not for any finite language, but for one given specific language A. At the same time, we do not strive for the polynomial algorithm of such a check: such a polynomiality would follow also the polynomiality of the verification algorithm of equivalence languages of two given nondeterministic finite automata.

The paper presents the formulations of a variant of the Zyu hypothesis using a petal nondeterministic automaton constructed according to some given finite language A. After that, the formulation of another variant of this hypothesis is given; it uses a canonical automaton equivalent to such a petal automaton. All variants of the hypothesis are illustrated by examples in which the same language is considered; at the same time, certainly, the examples are given only for one language, while hypotheses are formulated for the whole set of finite languages.

***Keywords*—formal languages, iterations of languages, morphisms, algorithms, nondeterministic finite automata, petal automata, determinization procedure, Zyu hypothesis.**

References

[1] Melnikov B., Melnikova A. *Infinite trees in the algorithm for checking the equivalence condition of iterations of finite languages. Part I* // International Journal of Open Information Technologies. – 2021. – Vol. 9. No. 4. – P. 1–11 (in Russian).

[2] Melnikov B., Melnikova A. *Infinite trees in the algorithm for checking the equivalence condition of iterations of finite languages. Part II* // International Journal of Open Information Technologies. – 2021. – Vol. 9. No. 5. – P. 1–11 (in Russian).

[3] Melnikov B. *Variants of finite automata, corresponding to infinite iterative morphism trees. Part I* // International Journal of Open Information Technologies. – 2021. – Vol. 9. No. 7. – P. 5–13 (in Russian).

[4] Melnikov B. *Variants of finite automata, corresponding to infinite iterative morphism trees. Part II* // International Journal of Open Information Technologies. – 2021. – Vol. 9. No. 10. – P. 1–8 (in Russian).

[5] Melnikov B. *The equality condition for infinite catenations of two sets of finite words* // International Journal of Foundation of Computer Science. – 1993. – Vol. 4. No. 3. – P. 267–274.

[6] Melnikov B. *Semi-lattices of the subsets of potential roots in the problems of the formal languages theory. Part I. Extracting the root from the language* // International Journal of Open Information Technologies. – 2022. – Vol. 10. No. 4. – P. 1–9 (in Russian).

[7] Melnikov B. *Semi-lattices of the subsets of potential roots in the problems of the formal languages theory. Part II. Constructing an inverse morphism* // International Journal of Open Information Technologies. – 2022. – Vol. 10. No. 5. – P. 1–8 (in Russian).

[8] Melnikov B. *Semi-lattices of the subsets of potential roots in the problems of the formal languages theory. Part III. The condition for the existence of a lattice* // International Journal of Open Information Technologies. – 2022. – Vol. 10. No. 7. – P. 1–9 (in Russian).

[9] Melnikov B. *On the complex of problems for the study of the necessary conditions for the equality of infinite iterations of finite languages* // International Journal of Open Information Technologies. – 2023. – Vol. 11. No. 1. – P. 1–12 (in Russian).

[10] Melnikov B. *Algorithm for checking equality of infinite iterations of finite languages* // Bulletin of the Moscow University. Series 15: Computational Mathematics and Cybernetics. – 1996. – No. 4. – P. 49–56 (in Russian).

[11] Melnikov B. *Some more on ω-finite automata and ω-regular languages. Part I: The main definitions and properties* // International Journal of Open Information Technologies. – 2020. – Vol. 8. No. 8. – P. 1–7.

[12] Lallement G. *Semigroups and Combinatorial Applications*. – NJ, Wiley & Sons, Inc. – 1979. – 376 p.

[13] Salomaa A. *Jewels of Formal Language Theory*. – Rockville, Maryland, Computer Science Press. – 1981. – 144 p.

[14] Melnikov B. *Subclasses of the context-free languages class (monograph)*. – Moscow, Moscow State University Ed. – 1995. – 174 p. – ISBN 5-211-03448-1 (in Russian).

[15] Melnikov B. *Regular languages and nondeterministic finite automata (monograph)*. – Moscow, Russian Social State University Ed. – 2018. – 179 p. – ISBN 978-5-7139-1355-7 (in Russian).

[16] Melnikov B., Dolgov V. *Simplified regular languages and a special equivalence relation on the class of regular languages. Part I* // International Journal of Open Information Technologies. – 2022. – Vol. 10. No. 9. – P. 12–20 (in Russian).

[17] Melnikov B., Dolgov V. *Simplified regular languages and a special equivalence relation on the class of regular languages. Part II* // International Journal of Open Information Technologies. – 2023. – Vol. 11. No. 1. – P. 13–19 (in Russian).

Boris MELNIKOV,
Professor of Shenzhen MSU – BIT University, China
(http://szmsubit.ru/),
email$_1$: bormel@smbu.edu.cn,
email$_2$: bf-melnikov@yandex.ru,
mathnet.ru: personid=27967,
elibrary.ru: authorid=15715,
scopus.com: authorId=55954040300,
ORCID: orcidID=0000-0002-6765-6800.

Aleksandra MELNIKOVA,
Associated Professor of
Dimitrovgrad Engineering and Technology Institute –
Branch of National Research Nuclear University "MEPhI"
(https://diti-mephi.ru/),
email: super-avahi@yandex.ru,
mathnet.ru: personid=148963,
elibrary.ru: authorid=143351,
scopus.com: authorId=6603567251,
ORCID: orcidID=0000-0002-1658-6857.