

О комплексе задач исследования необходимых условий равенства бесконечных итераций конечных языков

Б. Ф. Мельников

Аннотация—В нескольких предыдущих публикациях мы исследовали свойства специального бинарного отношения, которое иногда называли отношением эквивалентности в бесконечности. Для пары конечных языков для выполнения этого отношения необходимо и достаточно равенство бесконечных итераций этих языков.

Ранее мы рассматривали примеры применения этого отношения – причём как примеры, описывающие необходимые условия его выполнения, так и примеры его использования в различных областях теории формальных языков. Задачи, рассматриваемые в настоящей статье, можно назвать задачами исследования необходимых условий равенства бесконечных итераций конечных языков, построение на основе этих условий алгоритмов проверки принадлежности некоторого конечного языка множеству морфических образов расширенных максимальных префиксных кодов и алгоритмов построения соответствующих морфизмов, а также применение подобных алгоритмов в задачах теории формальных языков.

Основным предметом статьи является мотивация к рассмотрению подобных задач. И, по-видимому, главной «составляющей» такой мотивации является план доказательства возможности сведения равенства $P=NP$ к специальной гипотезе теории формальных языков, сформулированной ранее в одной из наших предыдущих публикаций. Стоит отметить, что первая версия этого плана была нами опубликована в 2011 г.; здесь приведена изменённая версия, причём со значительно бóльшим числом комментариев. Однако основная идея при этом та же самая – но здесь она уже оформлена конкретно, а решения многих вспомогательных подзадач, необходимых для осуществления такого плана, за прошедшее время уже опубликованы.

Кратко эту основную идею можно сформулировать так. По некоторому недетерминированному конечному автомату специальным образом строится пара конечных языков – и для этой пары показывается, что если бы была выполнена вышеупомянутая гипотеза, 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 в качестве определённого в наших предыдущих публикациях автомата NSPRI, 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 слова). Таким образом, выполнение вышеупомянутой гипотезы влечёт выполнение равенства $P=NP$.

Ключевые слова—формальные языки, итерации языков, бинарные отношения, морфизмы, инверсные морфизмы, алгоритмы, полиномиальные алгоритмы.

I. ВВЕДЕНИЕ

В статье мы продолжаем тематику публикаций [1], [2], [3], [4] – и здесь мы впервые подробно (полностью) описываем именно мотивацию к выполнению соответствующих этой теме задач.

Как было отмечено в [5],

«в ближайшем будущем мы предполагаем опубликовать специальную статью, посвящённую именно *мотивации* к рассмотрению задач, связанных с алгоритмами проверки выполнения бинарного отношения \Leftrightarrow » –

это и есть предмет настоящей статьи.

Во всех упомянутых выше статьях¹ мы исследовали свойства этого отношения \Leftrightarrow , впервые рассмотренного нами (в несколько иной интерпретации) ещё в 1993 г., в [6] – т.н. отношения эквивалентности в бесконечности. И, как мы также отмечали в [5], в дальнейших публикациях мы рассмотрели примеры применения этого отношения – причём эти примеры можно разделить на две группы:

- примеры, описывающие необходимые условия его выполнения;
- примеры его использования – в различных областях теории формальных языков, дискретной математики и абстрактной алгебры.

В связи с этим полное название настоящей статьи могло бы быть таким:

«О задачах исследования необходимых условий равенства бесконечных итераций конечных языков, построение на основе этих условий алгоритмов проверки принадлежности некоторого конечного языка множеству морфических образов расширенных максимальных префиксных кодов и алгоритмы построения соответствующих морфизмов, а также применение подобных алгоритмов в различных задачах теории формальных языков» –

но, конечно, такое «средневековое» название может кому-то показаться слишком длинным.

Специально отметим, что в статье мы не приводим новых алгоритмов, новых доказанных результатов и т.п.; её основной предмет – мотивация к рассмотрению подобных задач, т.е. к исследованию необходимых условий равенства бесконечных итераций двух конечных языков.

¹ А также во многих других. Большинство из них процитировано далее по тексту настоящей статьи.

При этом большая часть настоящей статьи посвящена именно описанию применения вариантов таких необходимых условий.

Итак, основным предметом статьи можно считать мотивацию к рассмотрению задач исследования необходимых условий равенства бесконечных итераций конечных языков. И, по-видимому, главной «составляющей» такой мотивации является план доказательства возможности сведения равенства $P=NP$ к специальной гипотезе теории формальных языков, сформулированной ранее в одной из наших предыдущих публикаций (и обозначаемой нами (\mathfrak{A})). Стоит отметить, что первая версия этого плана была нами опубликована в 2011 г.; здесь приведена изменённая версия, причём со значительно большим числом комментариев. Однако основная идея при этом та же самая – но здесь она уже оформлена конкретно, и решения многих вспомогательных подзадач, необходимых для осуществления такого плана, за прошедшее время уже опубликованы.

Кратко эту основную идею можно сформулировать следующим образом. По некоторому недетерминированному конечному автомату специальным образом строится пара конечных языков – и для этой пары показывается, что если бы была выполнена вышеупомянутая гипотеза, 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, 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 слова). Таким образом, осуществив кратко описанный здесь план, мы покажем, что выполнение вышеупомянутой гипотезы (\mathfrak{A}) влечёт выполнение равенства $P=NP$.

В конце введения отметим следующее. Мы всюду пишем «сведение равенства $P=NP$ к одной из гипотез теории формальных языков» (и т. п.) – но видимо ещё точнее было бы говорить «сведение гипотезы $P=NP$ к одной из гипотез теории формальных языков»; однако последний вариант, по-видимому, звучит хуже (из-за повторения слова). При этом мы помним, что почти каждую из рассматриваемых практических задач² тоже можно сформулировать именно как задачу теории формальных языков – см. [7], [8] и мн. др. – однако здесь речь идёт о формулировке задачи про задачи, можно сказать – «метазадачи».

Приведём содержание статьи по разделам. В разделе II мы рассматриваем необходимые обозначения. В разделе III приведены постановка основной задачи (определения эквивалентности в бесконечности итераций двух конечных языков) и несколько вариантов описания гипотез теории формальных языков, необходимых для плана

сведения равенства $P=NP$ к одной из таких гипотез. План такого сведения описан в разделе IV.

В разделе V приведены краткие описания других задач теории формальных языков, относящихся к той же самой (основной) предметной области; в разделе VI – описания «немного более далёких» от этой предметной области решавшихся и решаемых нами задач, но также имеющих к ней некоторое отношение. Раздел VII – заключение, в нём приведены возможные направления дальнейших работ по этой тематике.

II. ОСНОВНЫЕ ОБОЗНАЧЕНИЯ

Сначала повторим определения бинарного отношения \trianglelefteq и связанных с ним понятий – близко к [3], [5] и др.; при этом некоторая часть текста настоящего раздела повторена из [9].

Для заданного конечного алфавита Σ (иногда называемого нами «главным» либо «основным») и слова $u \in \Sigma^*$ над этим алфавитом:

- язык $\text{pref}(u)$ определяется как множество префиксов слова u (включая несобственный, само u);
- язык $\text{opref}(u)$ определяется как множество собственных префиксов слова u ;
- язык $\text{suff}(u)$ определяется как множество суффиксов слова u (включая несобственный, само u);
- язык $\text{osuff}(u)$ определяется как множество собственных суффиксов слова u .

Для заданного языка $A \subseteq \Sigma^*$

$$\text{pref}(A) = \bigcup_{u \in A} \text{pref}(u);$$

аналогичным образом определяются $\text{opref}(A)$, $\text{suff}(A)$ и $\text{osuff}(A)$.

Если для конечных языков A и B (в наших статьях они практически всегда рассматриваются над этим «основным» алфавитом Σ) выполнено условие

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

то будем писать $A \trianglelefteq B$ (либо $B \trianglerighteq A$). Если одновременно выполнены условия $A \trianglelefteq B$ и $A \trianglerighteq B$, мы будем писать $A \trianglelefteq\triangleright B$.

(Специально отметим ещё раз разницу в обозначениях: в наших последних статьях с одной стороны, и, с другой стороны, в [6] и нескольких статьях 1990-х годов. Конечно же, в настоящей статье мы будем применять только обозначения, приведённые в этом разделе.)

Следующее обозначение не было необходимым в [9] – однако нужно в настоящей статье, прежде всего – для материала, приведённого в разделе IV. В случае, когда уже известно, что условие $A \trianglerighteq B$ не выполнено, мы в случае выполнения условия $A \trianglelefteq B$ можем также писать $A \triangleleft B$ (либо $B \triangleright A$).

Далее считаем, что все рассматриваемые языки – непустые и не содержащие ε . (Если иного не сказано в их определениях – явно или неявно; а «неявно» – это, например, в случае задания языка как итерации некоторого другого.)

Для некоторого языка

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

² Задача коммивояжёра – один из очень многих примеров, в литературе описаны тысячи практических задач. Про каждую из них мы либо можем знать, что она принадлежит одному из классов P , NP и др., – либо не иметь такой информации.

мы рассматриваем алфавит

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

(причём если это не вызывает неоднозначностей, мы обычно пишем просто Δ), а для последнего алфавита – рассматриваем морфизм

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

задающийся следующим образом:

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

Если A не обладает свойством префикса, т.е. если

$$(\exists u, v \in A) (u \in \text{opref}(v)),$$

то определённый здесь морфизм также будем называть *непрефиксным*.

Для заданных конечного языка A , морфизма h_A и некоторого слова $u \in \Sigma^*$ рассмотрим *язык*

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

специально подчеркнём, что определённое понятие является *множеством*. Ту же самую конструкцию можно рассматривать и для некоторого языка (вместо слова u ; пусть это язык B) – причём нас будут интересовать только конечные языки. A именно:

$$h_A^{-1}(B) = \bigcup_{u \in B} h_A^{-1}(u).$$

В следующем определении, не связанном с предыдущим, «основной» алфавит – Δ . Над этим алфавитом множество максимальных префиксных кодов – как множество языков – во многих наших предыдущих работах обозначалось $\text{mp}(\Delta)$. При этом определение «одного» максимального префиксного кода – также как некоторого языка – мы приводить не будем; достаточно сказать, что это – (однозначный) код, который является максимальным и префиксным. См. некоторые комментарии, включающие определение кодового индекса, в [9].

A множество языков, каждый из которых *содержит* некоторый максимальный префиксный код *в качестве подмножества* (возможно, несобственного), обозначается $\text{mp}^+(\Delta)$. Таким образом,

$$\text{mp}(\Delta) \subset \text{mp}^+(\Delta),$$

причём, конечно, для любого алфавита включение собственное. Мы будем называть каждый из таких языков *расширенным максимальным префиксным кодом*.

Вернёмся к рассмотрению «основного» алфавита, т.е. Σ , а также морфизмов вида

$$h_A : \Delta_A^* \rightarrow \Sigma^*.$$

Пусть у нас есть некоторый язык

$$A_\Delta \in \text{mp}(\Delta);$$

тогда будем считать, что выполнено условие

$$h_A(A_\Delta) \in \text{mp}(A)$$

(мы таким образом определяем множество языков $\text{mp}(A)$ над рассматриваемым алфавитом Σ). Аналогично, для некоторого языка

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

будем считать, что

$$h_A(A_\Delta) \in \text{mp}^+(A).$$

Таким образом:

- $\text{mp}^+(\Delta)$ – это множество языков, каждый из которых содержит в качестве подмножества некоторый максимальный префиксный код над алфавитом Δ ;
- $\text{mp}^+(A)$ – это множество языков, каждый из которых является A -морфизмом некоторого языка из множества $\text{mp}^+(\Delta)$; т.е. каждый из таких языков – специальный морфизм некоторого расширенного максимального префиксного кода.

III. ПОСТАНОВКА ОСНОВНОЙ ЗАДАЧИ И ВАРИАНТЫ ОПИСАНИЯ ГИПОТЕЗ

В этом разделе мы также повторяем некоторую часть текста, приведённую в [5], [9].

В предыдущей статье [5] мы описали детерминированный полиномиальный алгоритм проверки выполнения условия морфического образа расширенного максимального префиксного кода – то есть проверки условия $A \in \text{mp}^+(D)$ для заданных непустых конечных языков $A, D \subseteq \Sigma^*$. Этот алгоритм можно считать одной из важных вспомогательных задач, необходимых для исследования сформулированного выше отношения \trianglelefteq на множестве непустых конечных языков.

Но, конечно, постановка задачи, приведённая в предыдущем абзаце, – очень краткая. Поэтому повторим (с некоторыми дополнениями) тот вариант постановки задачи, который был приведён в расширенной аннотации предыдущей статьи [5]. Такое более полное описание постановки основано на рассмотренных в предыдущем разделе настоящей статьи определениях максимальных префиксных кодов, морфизмов и инверсных морфизмов.

Итак, в наших предыдущих публикациях были рассмотрены бесконечные итерации конечных языков – как ω -языки. При этом бесконечные итерации двух заданных конечных языков A и B совпадают тогда и только тогда, когда для них выполняется описанное выше бинарное отношение \trianglelefteq :

$$A^\omega = B^\omega \quad \text{тогда и только тогда, когда} \quad A \trianglelefteq B.$$

Была сформулирована *гипотеза* о том, что *это возможно тогда и только тогда, когда оба рассматриваемых конечных языка могут быть получены с помощью алгоритма, состоящего из следующих 3 шагов:*

- 1) выбирается *некоторый*³ новый конечный алфавит;
- 2) над этим алфавитом рассматриваются *некоторые* два расширенных максимальных префиксных кода;
- 3) к этим двум расширенным максимальным префиксным кодам применяется *некоторый* (причём обязательно *один и тот же в обоих случаях*) морфизм – переводящий слова над новым алфавитом в слова над алфавитом исходным.

³ При кратком описании этого алгоритма слово «некоторый» употребляется часто – и выделяется курсивом; аналогично – слова «могут быть получены», см. выше. Это связано с тем, что мы *заранее не можем знать*, какой именно объект (какое именно отображение) должно применяться.

Алгоритмы определения (построения) таких конкретных объектов (конкретных отображений) – это, по большому счёту, и есть комплекс задач, рассматриваемых в наших публикациях по этой тематике.

При этом полученные морфические образы двух рассматриваемых расширенных максимальных префиксных кодов должны совпасть с двумя исходными конечными языками.

Теперь немного более подробно. В предыдущих работах нами было сформулировано *несколько эквивалентных вариантов* этой гипотезы: это эквивалентные между собой варианты первой гипотезы, обозначенные нами ранее, т. е. в [10, разд. XIV], следующим образом:

- (\mathfrak{A}') – этот вариант фактически совпадает с приведённым выше.
- а также (\mathfrak{A}) , (\mathfrak{A}'') и (\mathfrak{A}''') , см. далее.

Кроме них также была сформулирована другая гипотеза, представляющая собой более сильное утверждение⁴, о котором имеет смысл говорить в случае невыполнения гипотезы первой – эта вторая гипотеза была обозначена $(\mathfrak{A}\mathfrak{A})$, также см. далее.

Итак, кратко повторим формулировку этих гипотез (кроме (\mathfrak{A}')) – и отметим, что в [10] они были прокомментированы подробнее. Там же были приведены ссылки на возможные доказательства эквивалентности 4 вариантов первой гипотезы.

Гипотеза \mathfrak{A} . Для любых конечных языков $A, B \subseteq \Sigma^*$ выполнено следующее. Эквивалентность $A \trianglelefteq B$ 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

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

Гипотеза \mathfrak{A}'' . Сформулированное в [6, Th. 1] утверждение является верным – т. е., иными словами, сформулированное в [6] достаточное условие эквивалентности $A \trianglelefteq B$ является необходимым и достаточным.

Гипотеза \mathfrak{A}''' . Не существует пары

$$(\text{язык } D \subseteq \Sigma^*, \text{ слово } u \in \Sigma^*),$$

для которой $u \notin D^*$, причём пары такой, что:

- язык D – минимальный в своём классе эквивалентности по отношению \trianglelefteq ;
- $(D \cup \{u\}) \trianglelefteq D$.

Гипотеза $\mathfrak{A}\mathfrak{A}$. Считаем, что гипотеза (\mathfrak{A}) не выполняется. Однако при этом *не существует* конечного языка $D \subseteq \Sigma^*$, для которого выполнено следующее.

Во-первых, для этого D гипотеза (\mathfrak{A}) *не выполняется* для каждого из следующих слов⁵:

$$u_1, u_2, \dots, u_n.$$

⁴ В процитированной работе было сказано по-другому, «менее сильное утверждение» – однако, как ни странно, оба варианта верны!

- «Более сильное» (как мы сказали здесь) – это в том смысле, что не выполнена гипотеза (\mathfrak{A}) , но при этом *должно быть выполнено нечто ещё* (см. далее), т. е. некоторый дополнительный факт (поэтому и «более сильное»).
- А слова «менее сильное» (как мы сказали в прошлой публикации) – тоже можно объяснить: не выполнена (\mathfrak{A}) , но что-то «всё-таки выполненное» мы формулируем.

Короче говоря, здесь – «лингвистические трудности», неоднозначность.

⁵ Согласно приведённым выше предположениям, можно утверждать, что:

Будем здесь обозначать

$$\{u_1, u_2, \dots, u_n\} = F.$$

Во-вторых, для этих языков D и F существует некоторое непустое множество $G \subseteq D$, такое что

$$D \trianglelefteq D_{F-G}.$$

Вернёмся к подробному описанию постановки решаемой задачи. Нам необходимо проверить эквивалентность $A \trianglelefteq B$, для чего, среди прочего, необходимо найти *минимальный* язык D , такой что $A \trianglelefteq D$. Одна из ещё более простых подзадач – это проверка, может ли один язык быть получен с помощью алгоритма, подобного описанному выше как \mathfrak{A}' , применённого к некоторому *другому* языку⁶. И, поскольку недетерминированный алгоритм для такой задачи тривиален ([11, разд. XII] и др.), – то в [4] мы привели *детерминированный, причём полиномиальный*, алгоритм проверки выполнения этого условия.

IV. О СВЕДЕНИИ РАВЕНСТВА $P=NP$ К ГИПОТЕЗЕ ТЕОРИИ ФОРМАЛЬНЫХ ЯЗЫКОВ

Более точное название этого раздела могло бы быть таким: «О плане доказательства возможности сведения равенства $P=NP$ к одной гипотезе теории формальных языков» (имеется в виду (\mathfrak{A})). Стоит отметить, что *самая первая версия такого плана* была нами опубликована ещё в 2011 г., см. [12]; конечно же, здесь приведена существенно подкорректированная версия, причём со значительно бóльшим числом комментариев. Однако основная идея при этом та же самая – но здесь она уже оформлена конкретно, и решения многих вспомогательных подзадач, необходимых для осуществления такого плана, за прошедшее время уже опубликованы.

Итак, *пусть всюду далее* условие (\mathfrak{A}) выполнено. Зная, что оно выполнено, мы каким-то способом хотим определить, выполнено ли для некоторых конечных языков $A, B \subseteq \Sigma^*$ равенство $A \trianglelefteq B$. Опишем *два возможных пути получения ответа* на этот вопрос.

Первый путь.

Начала мы для каждого из этих двух языков в качестве вспомогательной задачи строим оптимальный инверсный морфизм; согласно гипотезе (\mathfrak{A}) ⁷, мы при этом должны получить совпадающие языки – т. е. один и тот же конечный язык $D \subseteq \Sigma^*$, такой что

$$A, B \in \text{tr}^+(D).$$

(Отметим, что обозначение D при этом имеет такой же смысл, который использовался в [6].)

Для этой цели мы сначала рассматриваем язык A (и «пока забываем» о языке B). Для этого языка A мы:

- во-первых, рассматриваем его потенциальные корни ([9], [10], [11]):

$$A' = \overline{\sqrt{A}} = \{u \in \Sigma^* \mid (\exists n \in \mathbb{N})(u^n \in A)\};$$

- выполнено не только неравенство $n \geq 1$, но и $n \geq 2$;
- ни одно из указанных далее слов не входит в язык D^* .

⁶ Если допускать возможность совпадения языков, то ответ на подобный вопрос очевиден.

⁷ Можно сказать – согласно «основному» её варианту, т. е. (\mathfrak{A}) , «без штрихов».

специально отметим, что равенство $n = 1$ при этом допускается;

- во-вторых напомним, что рассматриваемая нами задача в принципе может быть решена перебором множества всех подмножеств множества потенциальных корней⁸;
- в-третьих, рассматриваем специальное подмножество множества потенциальных корней – а именно те из них, которые *не представляются* в виде следующей нетривиальной конкатенации других его потенциальных корней:

$$A'' = \{u \in A' \mid (\nexists u_1, u_2, \dots, u_k \in A', k \geq 2) (u = u_1 u_2 \dots u_k)\};$$

- в-четвёртых заметим, что язык A'' может быть построен на основе языка A с помощью полиномиального алгоритма (это очевидно: например, несколько аналогичных алгоритмов мы применяли в качестве вспомогательных в [13]);
- в-пятых, как несложно видеть (например, на основе материала [9], либо просто на основе определения бинарного отношения \trianglelefteq), что $A'' \trianglelefteq A'$; в связи с этим будем далее рассматривать только те потенциальные корни, которые входят в A'' .

Теперь в полученном языке выберем некоторое слово $u \in A''$ и рассмотрим задачу определения, верно ли что

$$A'' \trianglelefteq A'' \setminus \{u\}. \quad (1)$$

Вследствие выполнения гипотезы (\mathfrak{X}) мы при одновременном с ней выполнении условия (1) получаем, что $u \in (A'')^*$, – что даёт тривиальный (причём полиномиальный) алгоритм проверки этого условия (1).

Поэтому, *последовательно* удаляя все такие слова u , для которых условие (1) не выполнено – т.е., иными словами, удаляя такие $u \in A''$, для которых

$$A'' \triangleright A'' \setminus \{u\}, –$$

мы получаем искомый язык D , минимальный в классе эквивалентности⁹, определяемом языком A . И, поскольку полиномиально одно такое удаление, то полиномиален и весь алгоритм построения требуемого языка D .

(Конечно здесь был приведён не общий алгоритм построения требуемого языка D с необходимыми строгими доказательствами – а только *схема* такого алгоритма. Мы предполагаем привести полное описание в одной из ближайших публикаций.)

Второй путь.

Начнём описание этого пути с информации об «обратной задаче»: в [13] мы показали возможность построения по двум конечным языкам (в наших обозначениях – $A, B \subseteq \Sigma^*$) недетерминированного конечного автомата (обозначавшегося $NSPRI(A, B)$), результат работы которого даёт ответ на вопрос, выполнено ли отношение $A \trianglelefteq B$: оно выполнено тогда и только тогда, когда языком автомата $NSPRI(A, B)$ является «полный» («универсальный») язык Σ^* . Однако в общем случае ответ на последний вопрос (т.е. совпадает ли язык некоторого

недетерминированного конечного автомата с языком Σ^*) является NP-полной задачей – и это даёт «косвенный аргумент» об NP-полноте и более интересной нам «прямой задаче».

А её («прямую задачу») сформулируем так. Сначала нам желательно формально описать ограничения, налагаемые на (недетерминированный) автомат $NSPRI$; такие ограничения действительно имеются (см. [3], [4], [14] и др.) – например, следующее. Согласно свойствам этого автомата, наличие любого слова $u \in \Sigma^*$ как обозначения некоторого состояния этого автомата влечёт наличие в качестве других состояний этого автомата любого суффикса u , т.е. любого слова, принадлежащего множеству $\text{suff}(u)$:

$$u \in Q \implies (\forall v \in \text{pref}(u)) (v \in Q); \quad (2)$$

«внутри» настоящего раздела назовём (2) ограничением (Ω).

Отметим, что возможны и другие ограничения на автомат $NSPRI$ (не связанные с (Ω)) – но отметим также, что нам нет необходимости строго доказывать тот факт, что какое-то конкретное множество ограничений даёт описание любого возможного автомата $NSPRI$: даже если этим ограничениям удовлетворяет большее множество автоматов, мы сможем завершить рассмотрение такой «прямой задачи». Однако, конечно, *наличие этих ограничений не должно давать возможность за полиномиальное время ответить на вопрос, определяет ли автомат, удовлетворяющий таким ограничениям, «полный» язык Σ^* – и (Ω)¹⁰ таковым ограничением является.*

После описания ограничений необходимо описать процедуру построения пары (A, B) – которая должна получаться на основе заданного автомата (потенциального $NSPRI$) за полиномиальное время. Для описания такой процедуры мы предполагаем привести в одной из следующих публикаций алгоритм для одной из возможных *подзадач* этой задачи – описания полиномиальной процедуры добавления перехода. Опишем такую подзадачу немного подробнее.

- Пусть для некоторых языков A и B соответствующий им автомат $NSPRI(A, B)$ *уже имеется*.
- Пусть, кроме того, задаётся некоторое значение функции переходов

$$\delta(q, a) \ni q', \quad (3)$$

которое в заданном автомате $NSPRI(A, B)$ *отсутствовало*.

- Тогда нужно указать алгоритм, который за полиномиальное время *модифицирует* исходный автомат $NSPRI(A, B)$ – т.е., например, добавляет некоторые слова в языки A и/или B – в результате чего должна получаться модификация исходного автомата $NSPRI(A, B)$, а именно автомат, в котором дополнительно ко всем уже имеющимся переходам присутствует ещё и переход (3); при этом возможно изменение языков A и/или B – но такое изменение тоже должно выполняться за полиномиальное время.

⁸ Понятно, что такой алгоритм – экспоненциальный.

⁹ Напомним, что такая «минимальность» может быть определена различными возможными способами – некоторые из них были описаны в наших предыдущих публикациях.

¹⁰ Точнее – множество ограничений, состоящее из единственного (Ω).

Итак, если мы имеем полиномиально-временной алгоритм построения пары языков (A, B) по заданному автомату NSPRI, то тогда мы для них (для этой пары языков) определяем, выполнено ли условие $A \trianglelefteq B$:

- за полиномиальное время (конечно, в случае выполнения гипотезы (\mathfrak{A}))
- согласно описанному выше «первому пути».

Поэтому мы – также за полиномиальное время в случае выполнения гипотезы (\mathfrak{A}) – отвечаем на вопрос, совпадает ли язык автомата NSPRI с Σ^* .

Таким образом, *в случае выполнения гипотезы (\mathfrak{A}) рассматриваемую задачу, принадлежащую классу NP, можно также отнести к классу P.*

Ещё раз напомним, что здесь приведён лишь *план* доказательства (возможности сведения равенства $P = NP$ к гипотезе теории формальных языков). Добавим также, что к такому плану, конечно же, мы должны добавить строгое доказательство NP-полноты рассматриваемой задачи. Мы предполагаем выполнить этот план в последующих публикациях.

Стоит также отметить следующее. Весь предполагаемый план предназначен для случая, рассматривающего *выполнение* гипотезы (\mathfrak{A}) ; однако мы ничего не утверждаем для *противного* случая – т.е. если мы *найдем контрпример* к (\mathfrak{A}) (достаточно просто доказать *существование* подобного контрпримера). Специально отметим, что мы в этом случае *не утверждаем* выполнения неравенства $P \neq NP$.

В заключение раздела отметим следующее. Здесь мы применяли алгоритм, рассмотренный ранее в [13]. Однако, как мы уже отмечали при описании основных обозначений, для действий, необходимых для алгоритмов из настоящей статьи, необходим ещё и алгоритм проверки выполнения условия $A \triangleleft B$: он, в частности, нужен для построения минимального языка D (минимального в своём классе эквивалентности по отношению \trianglelefteq). Однако отметим, что такой алгоритм (проверки выполнения условия \triangleleft) несложно строится путём небольшой модификации алгоритма из [13] – а подробности такой модификации мы приведём в одной из будущих статей, при рассмотрении деталей описанного в настоящем разделе плана.

V. ЕЩЁ ОБ ОСНОВНОЙ ПРЕДМЕТНОЙ ОБЛАСТИ

Итак, в настоящей статье мы продолжаем тематику, связанную со специальным бинарным отношением на множестве формальных языков (рассматриваемом в первую очередь на множестве итераций непустых конечных языков) – т.н. отношением эквивалентности в бесконечности. Ранее, кроме отношения \trianglelefteq мы также сформулировали более простое бинарное отношение на множестве языков – т.н. отношение покрытия, \triangleleft , двойное применение которого равносильно применению отношения \trianglelefteq . В предыдущих публикациях мы рассматривали алгоритм проверки выполнения отношения покрытия, а также определили вспомогательные объекты, используемые для доказательства корректности этого алгоритма. С одним из подобных объектов – а именно, с расширенным максимальным префиксным кодом, точнее, с бесконечным множеством таких кодов, – мы и работаем в настоящей статье.

Также в предыдущих публикациях (прежде всего – в [1], [2]) мы определили ещё один вид таких объектов, т.н. бесконечные итерационные деревья. После получения конкретного дерева мы объединяем эквивалентные его вершины (эквивалентность вершин рассматривается на основе определения таких деревьев) – фактически получая при этом детерминированный конечный автомат. Мы определили конкретный такой автомат для двух заданных конечных языков (подлежащих проверке на выполнение отношения \trianglelefteq) – т.н. автомат PRI; он, как уже было отмечено, *детерминированный*, определён на *множествах* слов, – и каждое из этих множеств является подмножеством множества префиксов второго из заданных языков¹¹. После этого мы определяем соответствующие такому автомату несколько вариантов специальных *недетерминированных* автоматов, фактически описывающих построение заданного итерационного дерева морфизма. Мы вводим совершенно иной объект – т.н. автомат NSPRI, – который также описывает построение итерационного дерева морфизма, но определён не на конечных множествах слов (конечных языках), а на (отдельных) словах. (Как уже было отмечено, про автоматы NSPRI и PRI см. [3], [4], [14].)

Название настоящей статьи начинается со слов «полиномиальный алгоритм» – поэтому скажем немного о полиномиальности другого, связанного алгоритма, описанного нами в [13]. Основная идея доказательства того факта, что алгоритм *построения конечного автомата для проверки равенства бесконечных итераций двух конечных языков* полиномиален, такова. Естественным является построение для этой задачи детерминированного автомата – каждое состояние которого описывает целое множество возможных префиксов, оставшихся от различных разложений слова – т.е. разложений морфизма итерации первого заданного конечного языка по словам второго заданного конечного языка. Однако при этом мы работаем с *множеством множеств* возможных префиксов – что делает невозможным полиномиальность соответствующего алгоритма. Поэтому мы строим *недетерминированный* автомат – определённый просто на множестве возможных префиксов¹². При этом возникает вопрос – когда именно этот автомат даёт положительный результат. Мы решаем эту проблему тем, что требуем, чтобы получающийся недетерминированный автомат являлся бы аналогом всюду определённого детерминированного автомата¹³.

(При этом условие того, что недетерминированный автомат является аналогом всюду определённого детерминированного автомата, *за полиномиальное время проверить невозможно* – см. подробнее [13]. Однако именно невозможность подобной проверки может дать доказательство невозможности проверки за полиномиальное время и условия выполнения отношения \trianglelefteq для двух заданных конечных языков.)

¹¹ Возможен также вариант, когда вместо множества префиксов используется множество суффиксов.

¹² Специально повторим, что *не* на множестве подмножеств таких возможных префиксов.

¹³ Определение такого «аналога» очевидно. Если говорить неформально – должны быть заполнены все клетки таблицы, описывающей граф переходов (transition function) рассматриваемого недетерминированного автомата.

К основной рассматриваемой нами предметной области относится не только сказанное выше в этом разделе, но и некоторые другие варианты использования отношения \Downarrow в задачах теории формальных языков. Выше уже были приведены ссылки на некоторые публикации, в которых это отношение:

- либо исследовалось,
- либо применялось в других задачах.

Все упомянутые выше наши статьи – за исключением [6] – были датированы несколькими последними годами. А здесь мы (после рассмотрения некоторой аналогии) дадим ссылки на существенно более ранние статьи по этой тематике – причём, конечно, отметим связь этих статей с основной рассматриваемой нами предметной областью.

- Аналогия заключается в том, что для очень многих задач (даже предметных областей) существует разница – и иногда очень большая – между их «алгебраической составляющей» и «алгоритмической составляющей».

В качестве первого, очень простого примера, приведём предметную область из «студенческих курсов» – [15] и др.¹⁴ Во многих программах, выполненных на логических и функциональных языках программирования, можно понять результаты работы, точно не зная, как система в действительности их нашла. Поэтому стоит различать два уровня смысла программы (например, на Прологе) – а именно, декларативный смысл и процедурный смысл.

- Декларативный смысл касается только отношений, определённых в программе. Таким образом, декларативный смысл определяет, *что* должно быть результатом работы программы.
- С другой стороны, процедурный смысл определяет ещё и *как* этот результат был получен, т. е. как отношения реально обрабатываются Пролог-системой.

В качестве конкретного примера в [15] приводится *задача построения остовного дерева графа*. И понятно, что в этой задаче «алгебраическая составляющая» почти совпадает с декларативным смыслом, а «алгоритмическая составляющая» – с процедурным смыслом.

Стоит упомянуть и наши публикации на эту тему¹⁵ [16], [17], в которых фактически были приведены:

- как алгоритмы, которые можно назвать «декларативными»,
- так и алгоритмы, которые можно назвать «процедурными», –

несмотря на то, что всё выполнялось на «процедурном» языке Си++.

Второй конкретный пример – существенно более сложная предметная область, причём, однако, её тоже можно считать относящейся к «студенческим курсам». Согласно [18], в *реляционной модели данных* присутствуют оба механизма – реляционная

алгебра и реляционное исчисление. Запрос, представленный на языке реляционной алгебры, может быть *вычислен* на основе выполнения элементарных алгебраических операций. А для формулы реляционного исчисления однозначная вычислительная интерпретация, вообще говоря, отсутствует – поэтому языки реляционного исчисления являются в большей степени непроцедурными, или *декларативными*¹⁶.

Третий конкретный пример – предметная область, которая, по-видимому, наиболее близка к нашей тематике. Аналогию рассмотренным парам формализмов (алгоритмов) представляет *пара (грамматики – автоматы)*¹⁷, [7], [19], [20], [21] и мн. др. Грамматика (её можно считать аналогом формулы реляционного исчисления) определяет язык – но не даёт ответа на вопрос, *как узнать*, принадлежит ли некоторое конкретное слово этому языку. А ответ даёт автомат (кстати, являющийся в нашем случае аналогом реляционной алгебры – а не реляционного исчисления) – который в большинстве случаев строится по грамматике.

Возвращаясь к нашей теме и немного всё упрощая, можно охарактеризовать наши первые публикации (1990-х годов) следующим образом. Статья [6] описывает «алгебраическую составляющую» этой задачи, а [22] – «алгоритмическую». В последующих же публикациях на эту тему¹⁸ мы старались «свести эти составляющие вместе».

- В этом пункте мы повторим небольшой текст из одной из наших предыдущих публикаций, [23]¹⁹. В некоторых подклассах класса контекстно-свободных языков разрешима проблема эквивалентности, см. [24], [25], [26] – в отличие от всего этого класса, [7], [19], [27]. Важно отметить, что мы при этом имеем в виду не «пресловутый» класс детерминированных контекстно-свободных языков, проблема эквивалентности для которого была сформулирована ещё в конце 1960-х (см., например, [27, разд. 4.2]) и впоследствии была решена: частичные решения см. в [28], [29], [30], а полное – в [31]. Здесь мы имеем в виду подклассы, *не совпадающие* с классом детерминированных контекстно-свободных языков. Однако с помощью этих подклассов могут быть описаны некоторые *реальные* языки программирования – см. примеры в [25], [26]. Для описания этих

¹⁶ Повторим, что текст взят – с небольшими изменениями и добавленным курсивом – из процитированного выше источника.

При этом очень интересно отметить следующий факт. Реляционная алгебра описывает алгоритмический подход, а реляционное исчисление – алгебраический! Согласно же их названиям – «ожидалось» ровно наоборот! (Тем более, что слово «исчисление» имеет тот же корень, что и «вычисление», – а последнее понятие можно считать синонимом алгоритма?)

Мы не находили подобного замечания в научной или учебной литературе.

¹⁷ Причём рассматривать такие пары можно как для 3-го («регулярного»), так для 2-го («контекстно-свободного») типа языков в иерархии Хомского.

¹⁸ В тех из них, где рассматривалось не применение описанного алгоритма, а его дальнейшее исследование.

¹⁹ Как и выше в подобных ситуациях – с небольшими изменениями и дополнениями.

¹⁴ Здесь мы «без кавычек» процитируем несколько мыслей из этой книги. Аналогично – немного ниже, про реляционную модель данных. В обоих случаях тексты этих книг нами немного изменены и дополнены.

¹⁵ Несмотря на простоту формулировки, эту тему мы считаем далеко не «закрытой»!

подклассов мы часто рассматриваем пары языков²⁰, удовлетворяющих рассматриваемому в настоящей статье отношению эквивалентности $A \trianglelefteq B$.

- Ещё одним примером применения основной предметной области в теории формальных языков могут быть названы уже упомянутые статьи: [23], а также, вероятно, и [12]. Как видно из названий, в статьях рассматривается связь итераций языков с недетерминированными конечными автоматами и максимальными префиксными кодами. Однако более подробно здесь про содержание этих статей мы писать не будем: на основе их материала и сделаны практически все вышедшие впоследствии (2016 г. и позднее) наши публикации на эту тему, причём многие из этих публикаций уже процитированы в настоящей статье по другим поводам.

VI. О НЕКОТОРЫХ ПРЕДМЕТНЫХ ОБЛАСТЯХ, СВЯЗАННЫХ С ОСНОВНОЙ

Кроме сказанного выше сто́ит отметить и связь рассматриваемой в статье задачи с недетерминированными конечными автоматами, которые являлись основным предметом многих других наших предыдущих публикаций. Прежде всего эта связь заключается в том, что задачи, в которых рассматриваются расширенные максимальные префиксные коды, необходимы, среди прочего, для ранее определённых и частично исследованных нами конечными автоматами PRI и NSPRI – см. процитированные выше статьи. Однако только этими публикациями указанная связь далеко не ограничивается: многие свойства недетерминированных конечных автоматов, рассмотренные нами в предыдущих публикациях, напрямую с автоматами PRI и NSPRI не связаны; при этом они могут быть полезными для рассмотрения последних. Приведём по этому поводу несколько конкретных ссылок на наши предыдущие работы – с указанием конкретных возможных связей со свойствами автоматов PRI и NSPRI.

- В статье [32] рассматривался, среди прочего, наш вариант алгоритма построения регулярного выражения по заданному недетерминированному конечному автомату (можно сказать – наш вариант доказательства теоремы Клини, [33] и мн. др.). Особенностью этого варианта можно назвать тот факт, что мы работаем в первую очередь *не* с состояниями автомата (как в доказательстве, приведённом в той же монографии [33]²¹), а с его дугами – помечая их регулярными выражениями²² и постепенно уменьшая количество дуг, принадлежащих такому модифицируемому автомату²³.

²⁰ Точнее, пары итераций этих языков: в процитированных в этом пункте статьях использовались немного иные обозначения.

²¹ Да и вообще – во всех известных нам публикациях на эту тему.

²² Первоначально каждая дуга помечена регулярным выражением, «являющемся» буквой заданного автомата (или, возможно, пустым словом ε – когда рассматриваются автоматы со множеством пометок дуг из $\Sigma \cup \{\varepsilon\}$).

Отметим также, что возможность пометки дуг регулярными выражениями является «стартом» для рассмотрения расширенных конечных автоматов (extended finite automata), про которые кратко будет сказано далее.

²³ Во всех известных нам доказательствах сами пометки дуг не меняются. Первоначально они помечены.

В наших построениях, приведённых в упомянутой статье [32], важную роль, конечно же, играет использование звезды Клини – которая применяется и при определении максимальных префиксных кодов, являющихся одним из основных предметов настоящей статьи.

При этом важно отметить, что к тематике, о которой говорится в данном пункте, близки также следующие три темы:

- про аппроксимацию полугрупп, [34] и др.;
- про циклы базисного конечного автомата и их соответствие циклам произвольного недетерминированного конечного автомата, определяющего рассматриваемый регулярный язык, [35] и др.;
- про расширенные конечные автоматы (extended finite automata), [36], [37], [38] –

которые (темы) на первый взгляд кажутся мало связанными между собой; показать такую возможную связь может настоящая статья. О каждой из этих тем мы немного подробнее упомянем в заключении (раздел VII).

- В паре статей [39], [40]²⁴ мы рассмотрели свойства базисного автомата для заданного регулярного языка, связанные с образующимися в этом языке циклами; из сказанного здесь понятна возможная связь этой темы с расширенными максимальными префиксными кодами: исследование равенства пометок двух таких циклов может быть сведено к исследованию равенства морфизмов двух расширенных максимальных префиксных кодов.

Отметим также, что именно на основе этой пары статей мы впоследствии получили доказательства некоторых утверждений в следующих темах:

- построение любого автомата для заданного регулярного языка на основе базисного автомата для этого языка, [41]²⁵;
- алгоритмы построения автомата *СМ*, о котором будет сказано далее.

- Упомянутый автомат *СМ*, алгоритмы его построения и доказательство его совпадения с универсальным автоматом [42], [43], [44] (с точностью до переобозначения состояний) – всё это было нами рассмотрено в [45], [46] (см. также [47]).
- Кроме того, рассмотрение некоторых *алгебраических аспектов* тематики, связанной с выполнением бинарного отношения \trianglelefteq , было начато в [48]. Отметим, что это тема далеко не закрыта: в этой статье подмоноиды супермоноида свободного моноида классифицируются «с точки зрения префиксов», но при этом ещё возможно:

- расширить (дополнить) дакую классификацию;
- одновременно с ней – рассматривать ещё и классификацию подмоноидов «с точки зрения суффиксов».

²⁴ Несмотря на такой порядок опубликования, читать эти статьи желательно «в порядке обратном»: сначала про возможные дуги автомата для заданного регулярного языка – и уже потом про зависящие от них свойства базисного конечного автомата.

²⁵ Сто́ит, однако, отметить, что в этой статье были приведены *только* схемы алгоритма и соответствующего доказательства; полные варианты мы предполагаем опубликовать в будущем.

Обобщая всё это, можно сказать, что приведённый в [5] *практический* алгоритм – полиномиальный алгоритм проверки выполнения условия морфического образа расширенного максимального префиксного кода – может быть полезен для многих *теоретических* результатов, рассматриваемых в тех работах. Конкретно, многие приведённые там «чисто теоретические» алгоритмы ранее рассматривались без какой-либо оценки их сложности – поскольку она:

- либо – согласно тематике статей – просто не требовалась;
- либо была экспоненциальной – обычно это происходит из-за необходимости построения эквивалентного канонического автомата; при этом отметим, что подобные построения часто бывают необходимы и в тех случаях, когда явно они не указаны²⁶

Более того, в процитированных статьях легко отыскать ситуации, в которых сложность рассматриваемых алгоритмов составляет

$$\sim 2^{2^n},$$

где n – число состояний исходного автомата.

Теперь же, т. е. после публикации материала настоящей статьи, эту сложность алгоритмов можно:

- во-первых, оценить;
- и, во-вторых, иногда улучшить.

Однако, по-видимому, бóльшие подробности выходят за рамки рассматриваемой здесь тематики.

VII. ЗАКЛЮЧЕНИЕ

Итак, в настоящей статье мы рассмотрели:

- задачи исследования необходимых условий равенства бесконечных итераций конечных языков,
- построение (на основе этих условий) алгоритмов проверки принадлежности некоторого конечного языка множеству морфических образов расширенных максимальных префиксных кодов,
- алгоритмы построения соответствующих морфизмов,
- а также применение этих алгоритмов в различных задачах теории формальных языков.

Ещё раз отметим, что в статье мы не приводили новые алгоритмы, теоремы и т. п.; основной предмет статьи – мотивация к рассмотрению подобных задач.

Как обычно, приведём возможные направления дальнейших работ по этой тематике.

На основе материала статьи понятно, что основной темой является будущая работа с планом доказательства возможности сведения равенства $P=NP$ к одной гипотезе теории формальных языков – т. е. доведения этого плана, схематично описанного в разделе IV, до строгого изложения. Однако здесь мы имеем в виду сопутствующие подзадачи – которые необходимо решить в первую очередь.

Необходимо описать полиномиальный алгоритм, решающий задачу построения языка $h_A^{-1}(B)$, т. е. алгоритм, строящий язык – инверсный морфизм для некоторого

²⁶ Например, такие эквивалентные канонические автоматы необходимо строить при применении в алгоритмах либо функций разметки φ_K^n и φ_K^{out} , либо бинарного отношения $\# \dots$

заданного конечного языка. (Это была «более сложная задача», названная так в сноске 8 статьи [5].)

Не менее важно окончание работы над задачей построения *оптимального* инверсного морфизма. См. по этому поводу мотивацию, приведённую в [9], [10], [11]. И важно отметить, что, в отличие от материала статьи [5], при решении общей задачи построения оптимального инверсного морфизма язык B не задан.

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] Мельников Б., Мельникова А. *Полиномиальный алгоритм проверки выполнения условия морфического образа расширенного максимального префиксного кода* // International Journal of Open Information Technologies. – 2022. – Vol. 10. No. 12. – P. 1–9.
- [6] 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.
- [7] Хопкрофт Дж., Мотвани Р., Ульман Дж. *Введение в теорию автоматов, языков и вычислений*. М., Вильямс. – 2008. – 528 с.
- [8] Громкович Ю. *Теоретическая информатика. Введение в теорию автоматов, теорию вычислимости, теорию сложности, теорию алгоритмов, рандомизацию, теорию связи и криптографию*. СПб., БХВ-Петербург. – 2010. – 336 с.
- [9] 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.
- [10] 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.
- [11] 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.
- [12] Алексеева А., Мельников Б. *Итерации конечных и бесконечных языков и недетерминированные конечные автоматы* // Вектор науки Тольяттинского государственного университета. – 2011. – № 3 (17). – С. 30–33.
- [13] Мельников Б., Мельникова А. *Полиномиальный алгоритм построения конечного автомата для проверки равенства бесконечных итераций двух конечных языков* // International Journal of Open Information Technologies. – 2021. – Vol. 9. No. 11. – P. 1–10.
- [14] Абрамян М., Мельников Б. *Алгоритмы преобразования конечных автоматов, соответствующих бесконечным итерационным деревьям* // Современные информационные технологии и ИТ-образование. – 2021. – Том 17. № 1. – С. 13–23.
- [15] Братко И. *Алгоритмы искусственного интеллекта на языке Prolog*. – М., Вильямс. – 2004. – 604 с.

- [16] Мельников Б., Терентьева Ю. *Построение оптимального остовного дерева как инструмент для обеспечения устойчивости сети связи* // Известия высших учебных заведений. Поволжский регион. Технические науки. – 2021. – № 1 (57). – С. 36–54.
- [17] Melnikov B., Terentyeva Y. *Building communication networks: on the application of the Kruskal's algorithm in the problems of large dimensions* // IOP Conference Series: Materials Science and Engineering. – 2021. С. 12089.
- [18] Кузнецов С. *Базы данных*. – М., Изд-во ВМК МГУ. – 2020. – 255 с.
- [19] Ахо А., Ульман Дж. *Теория синтаксического анализа, перевода и компиляции. Т. 1*. М., Мир. – 1978. – 613 с.
- [20] Мельников Б. *Подклассы класса контекстно-свободных языков (монография)*. – М., Изд-во Московского университета. – 1995. – 174 с. – ISBN 5-211-03448-1.
- [21] 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.
- [22] Мельников Б. *Алгоритм проверки равенства бесконечных итераций конечных языков* // Вестник Московского университета. Серия 15: Вычислительная математика и кибернетика. – 1996. – № 4. – С. 49–56.
- [23] Корабельщикова С., Мельников Б. *Итерации языков и максимальные префиксные коды* // Вестник Воронежского государственного университета. Серия: Физика. Математика. – 2015. – № 2. – С. 106–120.
- [24] Мельников Б. *Некоторые следствия условия эквивалентности однозначных скобочных грамматик* // Вестник Московского университета. Серия 15: Вычислительная математика и кибернетика. – 1991. – № 10. – С. 51–54.
- [25] Дубасова О., Мельников Б. *Об одном расширении класса контекстно-свободных языков* // Программирование (Известия РАН). – 1995. – № 6. – С. 46–54.
- [26] Melnikov B., Kashlakova E. *Some grammatical structures of programming languages as simple bracketed languages* // Informatica (Lithuanian Academy of Sciences). – 2000. – Vol. 11. No. 4. – P. 441–454.
- [27] Гинзбург С. *Математическая теория контекстно-свободных языков*. М., Мир. – 1970. – 326 с.
- [28] Станевичене Л. *Об одном средстве исследования бесконтактных языков* // Кибернетика. – 1989. – № 4. – С. 135–136.
- [29] Staneviciene L. *D-graphs in context-free language theory* // Informatica (Lithuanian Academy of Sciences). – 1997. – Vol. 8. No. 1. – P. 43–56.
- [30] Мейтус В. *Разрешимость проблемы эквивалентности детерминированных магазинных автоматов* // Кибернетика и системный анализ. – 1992. – № 5. – С. 20–45.
- [31] Sénizergues G. *$L(A) = L(B)$? decidability results from complete formal systems* // Theoretical computer science. – 2001. – Vol. 251. No. 1–2. – P. 1–166.
- [32] Melnikov B., Vakhitova A. *Some more on the finite automata* // The Korean Journal of Computational and Applied Mathematics (Journal of Computational and Applied Mathematics). – 1998. – Vol. 5, No. 3. – P. 495–505.
- [33] Саломаа А. *Жемчужины теории формальных языков*. – М., Мир. – 1986. – 159 с.
- [34] Данг В. В., Корабельщикова С., Мельников Б. *О задаче нахождения минимальной полугруппы аппроксимации* // Известия высших учебных заведений. Поволжский регион. Физико-математические науки. – 2015. – № 3 (35). – С. 88–99.
- [35] Долгов В., Мельников Б., Мельникова А. *Циклы графа переходов базисного конечного автомата и связанные вопросы* // Вестник Воронежского государственного университета. Серия: Физика. Математика. – 2016. – № 4. – С. 95–111.
- [36] Melnikov B. *Extended nondeterministic finite automata* // Fundamenta Informaticae. – 2010. – Vol. 104, No. 3. – P. 255–265.
- [37] Мельников Б., Мельникова А. *Расширенный базисный конечный автомат. Часть I. Основные определения* // International Journal of Open Information Technologies. – 2019. – Vol. 7, No. 10. – P. 1–8.
- [38] Мельников Б., Мельникова А. *Расширенный базисный конечный автомат. Часть II. Описание вспомогательных языков и некоторые свойства* // International Journal of Open Information Technologies. – 2020. – Vol. 8, No. 5. – P. 1–7.
- [39] Melnikov B., Melnikova A. *Some properties of the basis finite automaton* // The Korean Journal of Computational and Applied Mathematics (Journal of Computational and Applied Mathematics). – 2002. – Vol. 9, No. 1. – P. 135–150.
- [40] Melnikov B., Sciarini-Guryanova N. *Possible edges of a finite automaton defining a given regular language* // The Korean Journal of Computational and Applied Mathematics (Journal of Computational and Applied Mathematics). – 2002. – Vol. 9, No. 2. – P. 475–485.
- [41] Мельников Б., Сайфуллина М. *О некоторых алгоритмах эквивалентного преобразования недетерминированных конечных автоматов* // Известия высших учебных заведений. Математика. – 2009. – № 4. – С. 67–72.
- [42] Kameda T., Weiner P. *On the state minimization of nondeterministic finite automata* // IEEE Transactions on Computers. – 1970. – Vol. C-19, No. 7. – P. 617–627.
- [43] Polák L. *Minimalizations of NFA using the universal automaton* // International Journal of Foundation of Computer Science. – 2005. – Vol. 16, No. 5. – P. 999–1010.
- [44] Lombardy S., Sakarovitch J. *The universal automaton* // Logic and Automata. Amsterdam University Press. – 2008. – P. 457–504.
- [45] Долгов В., Мельников Б. *Построение универсального конечного автомата. I. От теории к практическим алгоритмам* // Вестник Воронежского государственного университета. Серия: Физика. Математика. – 2013. – № 2. – С. 173–181.
- [46] Долгов В., Мельников Б. *Построение универсального конечного автомата. II. Примеры работы алгоритмов* // Вестник Воронежского государственного университета. Серия: Физика. Математика. – 2014. – № 1. – С. 78–85.
- [47] Долгов В., Мельников Б. *Об алгоритмах автоматического построения Ватерлоо-подобных конечных автоматов на основе полных автоматов* // Эвристические алгоритмы и распределенные вычисления. – 2014. – Т. 1. № 4. – С. 24–45.
- [48] Мельников Б. *Описание специальных подмоноидов глобального надмоноида свободно моноида* // Известия высших учебных заведений. Математика. – 2004. – № 3. – С. 46–56.

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

On the complex of problems for the study of the necessary conditions for the equality of infinite iterations of finite languages

Boris Melnikov

Abstract—In some previous publications, we investigated the properties of a special binary relation, which was sometimes called the equivalence relation at infinity. For a pair of finite languages, the equality of infinite iterations of these languages is necessary and sufficient to fulfill this relationship.

Earlier, we considered examples of the application of this relation: both examples describing the necessary conditions for its implementation, and examples of its use in various fields of formal language theory. The problems considered in this paper can be called the tasks of investigating the necessary conditions for the equality of infinite iterations of finite languages, building algorithms based on these conditions to verify that a finite language belongs to a set of morphic images of extended maximal prefix codes and algorithms for constructing corresponding morphisms, as well as the use of such algorithms in some problems of the theory of formal languages.

The main subject of the paper is motivation to consider such tasks. And, apparently, the main “component” of such motivation is the plan of proving the possibility of reducing the equality $P=NP$ to the special hypothesis of the theory of formal languages formulated earlier in one of our previous publications. It is worth noting that the first version of this plan was published by us in 2011; here is an amended version, and with a much larger number of comments. However, the basic idea is the same, but here it has already been formalized specifically, and the solutions to many auxiliary subproblems necessary for the implementation of such a plan have already been published over the past time.

Briefly, this basic idea can be formulated as follows. For some nondeterministic finite automaton, a pair of finite languages is constructed in a special way, and for this pair it is shown that if the above hypothesis were fulfilled, then there would be an algorithm for checking the fulfillment of the equivalence relation at infinity in polynomial time. But, on the other hand, taking an arbitrary nondeterministic finite automaton and considering it as an NSPRI automaton defined in our previous publications, we could construct a pair of finite languages corresponding to this automaton in polynomial time; and this pair satisfies the equivalence relation at infinity if and only if the language of the NSPRI automaton coincides with the universal language over a given alphabet (i.e., a language containing all possible words). Thus, the fulfillment of the above hypothesis entails the fulfillment of the equality $P=NP$.

Keywords—formal languages, iterations of languages, binary relations, morphisms, inverse morphisms, algorithms, polynomial algorithms.

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., Melnikova A. *A polynomial algorithm for checking the fulfillment of the condition of the morphic image of the extended maximal prefix code* // International Journal of Open Information Technologies. – 2022. – Vol. 10. No. 12. – P. 1–9 (in Russian).
- [6] Melnikov B. *The equality condition for infinite concatenations of two sets of finite words* // International Journal of Foundation of Computer Science. – 1993. – Vol. 4. No. 3. – P. 267–274.
- [7] Hopcroft J., Motwani R., Ullman J. *Introduction to Automata Theory, Languages, and Computation*. Massachusetts, Addison-Wesley Publishing Company Reading. – 2006. – 550 p.
- [8] Hromkovič J. *Introduction to Automata, Computability, Complexity, Algorithmics, Randomization, Communication, and Cryptography*. Berlin, Springer. – 2004. – 318 p.
- [9] 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).
- [10] 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).
- [11] 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).
- [12] Alekseeva A., Melnikov B. *Iterations of finite and infinite languages and nondeterministic finite automata* // Vector of Science of Togliatti State University. – 2011. – No. 3 (17). – P. 30–33 (in Russian).
- [13] Melnikov B., Melnikova A. *A polynomial algorithm for constructing a finite automaton for checking the equality of infinite iterations of two finite languages* // International Journal of Open Information Technologies. – 2021. – Vol. 9. No. 11. – P. 1–10 (in Russian).
- [14] Abramyan M., Melnikov B. *Algorithms of transformation of finite automata, corresponding to infinite iterative trees* // Modern information technologies and IT education. – 2021. – Vol. 17. No. 1. – P. 13–23 (in Russian).
- [15] Bratko I. *Prolog Programming for Artificial Intelligence*. – London, Pearson. – 2012. – 696 p.
- [16] Melnikov B., Terentyeva Y. *Building an optimal spanning tree as a tool for ensuring the stability of the communication*

- network* // News of higher educational institutions. Volga region. Technical sciences. – 2021. – No. 1 (57). – P. 36–54 (in Russian).
- [17] Melnikov B., Terentyeva Y. *Building communication networks: on the application of the Kruskal's algorithm in the problems of large dimensions* // IOP Conference Series: Materials Science and Engineering. – 2021. C. 12089.
- [18] Kuznetsov S. *Databases*. – Moscow, Edition of Department of Computational Mathematics and Cybernetics of Moscow State University. – 2020. – 255 p. (in Russian).
- [19] Aho A., Ullman J. *The Theory of Parsing, Translation, and Compiling. Vol. 1: Parsing*. NJ: Prentice Hall, 1972. – 560 p.
- [20] 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).
- [21] 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).
- [22] 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).
- [23] Korabelshchikova S., Melnikov B. *Iterations of languages and maximum prefix codes* // Bulletin of the Voronezh State University. Series: Physics. Mathematics. – 2015. – No. 2. – P. 106–120 (in Russian).
- [24] Melnikov B. *Some consequences of the equivalence condition of unambiguous bracketed grammars* // Bulletin of the Moscow University. Series 15: Computational Mathematics and Cybernetics. – 1991. – No. 10. – P. 51–54 (in Russian).
- [25] Dubasova O., Melnikov B. *On an extension of the class of context-free languages* // Programming and Computer Software (Russian Academy of Sciences Ed.). – 1995. – No. 6. – P. 46–54 (in Russian).
- [26] Melnikov B., Kashlakova E. *Some grammatical structures of programming languages as simple bracketed languages* // Informatica (Lithuanian Academy of Sciences). – 2000. – Vol. 11. No. 4. – P. 441–454.
- [27] Ginsburg S. *The mathematical theory of context free languages*. NY, McGraw-Hill Book Company. – 1966. – 232 p.
- [28] Stanevicienė L. *On a tool for the study of context-free languages* // Cybernetics. – 1989. – No. 4. – P. 135–136 (in Russian).
- [29] Stanevicienė L. *D-graphs in context-free language theory* // Informatica (Lithuanian Academy of Sciences). – 1997. – Vol. 8. No. 1. – P. 43–56.
- [30] Meytus V. *The solvability of the equivalence problem of deterministic push-down automata* // Cybernetics and System Analysis. – 1992. – No. 5. – P. 20–45 (in Russian).
- [31] Sénizergues G. *$L(A) = L(B)$? decidability results from complete formal systems* // Theoretical computer science. – 2001. – Vol. 251. No. 1–2. – P. 1–166.
- [32] Melnikov B., Vakhitova A. *Some more on the finite automata* // The Korean Journal of Computational and Applied Mathematics (Journal of Computational and Applied Mathematics). – 1998. – Vol. 5, No. 3. – P. 495–505.
- [33] Salomaa A. *Jewels of Formal Language Theory*. Computer Science Press, Rockville, Maryland. – 1981. – 144 p.
- [34] Dang V. V., Korabelshchikova S., Melnikov B. *About the problem of finding minimal approximation semigroup* // News of higher educational institutions. Volga region. Physical and Mathematical sciences. – 2015. – No. 3 (35). – P. 88–99 (in Russian).
- [35] Dolgov V., Melnikov B., Melnikova A. *Cycles of the transition graph of a basic finite automaton and related issues* // Bulletin of the Voronezh State University. Series: Physics. Mathematics. – 2016. – No. 4. – P. 95–111 (in Russian).
- [36] Melnikov B. *Extended nondeterministic finite automata* // Fundamenta Informaticae. – 2010. – Vol. 104, No. 3. – P. 255–265.
- [37] Melnikov B., Melnikova A. *Extended basis finite automaton. Part I. Basic definitions* // International Journal of Open Information Technologies. – 2019. – Vol. 7, No. 10. – P. 1–8 (in Russian).
- [38] Melnikov B., Melnikova A. *Extended basis finite automaton. Part II. The description of auxiliary languages and some properties* // International Journal of Open Information Technologies. – 2020. – Vol. 8, No. 5. – P. 1–7 (in Russian).
- [39] Melnikov B., Melnikova A. *Some properties of the basis finite automaton* // The Korean Journal of Computational and Applied Mathematics (Journal of Computational and Applied Mathematics). – 2002. – Vol. 9, No. 1. – P. 135–150.
- [40] Melnikov B., Sciarini-Guryanova N. *Possible edges of a finite automaton defining a given regular language* // The Korean Journal of Computational and Applied Mathematics (Journal of Computational and Applied Mathematics). – 2002. – Vol. 9, No. 2. – P. 475–485.
- [41] Melnikov B., Sayfullina M. *On some algorithms for equivalent transformation of nondeterministic finite automata* // News of higher educational institutions. Mathematics. – 2009. – No. 4. – P. 67–72 (in Russian).
- [42] Kameda T., Weiner P. *On the state minimization of nondeterministic finite automata* // IEEE Transactions on Computers. – 1970. – Vol. C-19, No. 7. – P. 617–627.
- [43] Polák L. *Minimalizations of NFA using the universal automaton* // International Journal of Foundation of Computer Science. – 2005. – Vol. 16, No. 5. – P. 999–1010.
- [44] Lombardy S., Sakarovitch J. *The universal automaton* // Logic and Automata. Amsterdam University Press. – 2008. – P. 457–504.
- [45] Dolgov V., Melnikov B. *Building a universal finite automaton. I. From the theory to the practical algorithms* // Bulletin of the Voronezh State University. Series: Physics. Mathematics. – 2013. – No. 2. – P. 173–181 (in Russian).
- [46] Dolgov V., Melnikov B. *Building a universal finite automaton. II. Examples of how algorithms work* // Bulletin of the Voronezh State University. Series: Physics. Mathematics. – 2014. – No. 1. – P. 78–85 (in Russian).
- [47] Dolgov V., Melnikov B. *On the automatic construction algorithms of Waterloo-like finite automata based on complete automata* // Heuristic algorithms and distributed computing. – 2014. – Vol. 1. No. 4. – P. 24–45 (in Russian).
- [48] Melnikov B. *Description of special submonoids of the global supermonoid of the free monoid* // News of higher educational institutions. Mathematics. – 2004. – No. 3. – P. 46–56 (in Russian).

Boris MELNIKOV,
 Professor of Shenzhen MSU–BIT University, China
 (<http://szmsubit.ru/>),
 email₁: 6620180056@smbu.edu.cn,
 email₂: bf-melnikov@yandex.ru,
 mathnet.ru: personid=27967,
 elibrary.ru: authorid=15715,
 scopus.com: authorId=55954040300,
 ORCID: orcidID=0000-0002-6765-6800.