УДК 512

Алгоритмические проблемы и ограничения на сложность вывода

В.А. Ганов[1], Р.В. Дегтерева[2]

[1] Алтайский государственный университет (Барнаул, Россия)
[2] Алтайский государственный технический университет им. И.И. Ползунова (Барнаул, Россия)

Algorithmic Problems and Restrictions on Inference Complexity

V.A. Ganov, R.V. Degtereva

[1] Altai State University (Barnaul, Russia)
[2] Polzunov Altai State Technical University (Barnaul, Russia)

Авторы продолжают свои исследования логических сетей над базисом конечных автоматов. Вводится специальный базис конечных автоматов и определяются словарные операторы, реализуемые логическими сетями над этим базисом. Программы автоматов связаны с правилами вывода известного исчисления Поста. Главная особенность этого исчисления в том, что в нем проблема эквивалентности слов не является алгоритмически разрешимой. Отсюда следует, что множество операторов, реализуемых данными сетями, также не является алгоритмически разрешимым.

Далее определяются правильно организованные логические сети, и показывается, что только автоматы таких сетей работают правильно. Сложность вывода определяется следующим образом. Выделяются так называемые основные автоматы, которые непосредственно моделируют правила вывода слов в данном исчислении. Сложностью вывода называется число основных автоматов, содержащихся в вычислительной модели этого вывода. Доказывается, что при фиксированном ограничении множество операторов, реализуемых выделенными сетями, является алгоритмически разрешимым.

Каждый реализуемый оператор связан с некоторым словом алфавита A_0, выводимом в L_0. Поэтому для данного оператора, соответствующего слову T, можно организовать поиск вывода этого слова путем перебора всех возможных выводов. И если искомый вывод короткий, то по теореме 10 он будет найден. А если этот вывод очень длинный или не существует, то этот метод не поможет.

Ключевые слова: конечные автоматы Мура, логические сети, вывод слова в исчислении Бэра.

In the paper, the authors continue their study of logical nets on the basis of finite automata. A special basis of finite automata and dictionary operators implemented by logical nets on this basis are introduced. Automata programs are associated with inference rules of well-known Post calculus. The key feature of the Post calculus is that words equivalency problem is algorithmically unsolvable. Therefore, a set of operators implemented by these nets are also algorithmically unsolvable.

Then properly constructed logical nets are identified, and only these nets on the basis of finite automata are shown to operate correctly. The inference complexity is defined in the following way: the so-called basic automata are identified that directly simulate inference rules for words in a specific calculus. Then the inference complexity is taken to be equal to the number of automata in the computational model of this inference. It is proved that for fixed restrictions a set of operators implemented by the previously identified nets is algorithmically solvable.

Every implemented operator is associated with a word of an alphabet A_0, inferred in L_0. Therefore, for a given operator with a corresponding word T it is possible to infer the word T by exhausting all possible inferences. If the desired inference is short then in accordance with the Theorem 10 it can be obtained. Otherwise, if the desired inference is too long or non-existent then the proposed technique will fail.

Keywords: Moore finite automata, logical network, inference of a word in the Baer calculus.

DOI 10.14258/izvasu(2015)1.2-17

Нормальные алгоритмы Поста-Маркова появились позже, чем понятие машины Тьюринга или рекурсивной функции, и их изучение велось обособленно. Поэтому иногда возникают вопросы: действительно ли эти алгоритмы эквивалентны рекурсивным функциям? И если это так, то в каком смысле они эквивалентны? При этом остается какая-то неясность. Действительно ли неразрешимая проблема для алгоритмов Маркова и неразрешимая проблема в теории рекурсивных функций — это две версии одной и той же проблемы, и что их эквивалентность связана с какими-то молчаливо подразумеваемыми предположениями?

В данной работе эти вопросы рассматриваются в связи с понятием выводимости слов в нормальном исчислении Поста L_4 из [1] и понятием реализации словарных операторов логическими сетями над специальным базисом конечных автоматов из [2]. Работа этих автоматов непосредственно связана с правилами вывода исчисления L_4. Доказывается, что класс операторов, реализуемых такими сетями, не является алгоритмически разрешимым. Но доказательство этого факта опирается на известную теорему Поста, согласно которой множество слов, выводимых в L_4 из слова *feam*, не является алгоритмически разрешимым. При этом доказательство этой теоремы осуществляется в рамках формальной арифметики, и это обстоятельство не устраивает авторов данной работы. Дело в том, что в работе [3] доказано, что формальная арифметика является ω-противоречивой, и, в частности, в ней выводимы интуитивно ложные утверждения. При этом противоречия возникают на достаточно высоком уровне, когда в рассуждения включаются формулы, в которых значениями переменных являются геделевские номера формул. В данной работе подобные формулы не рассматриваются, и исследования осуществляются в языке конечных автоматов.

1. Нормальное исчисление Поста.

Определение 1. *Нормальное исчисление Поста* L_4 — это всевозможные слова алфавита $A_0 = \{a,b,c,d,e,f,g,h,i,j,k,l,m,n\}$, над которыми можно производить определяемые ниже допустимые действия. Пусть A, B — фиксированные слова алфавита A_0. Запись $A \Leftrightarrow B$ называется *соотношением слов* A и B. Тогда этому соотношению соответствуют следующие два вида действий V_1 и V_2, допустимых относительно $A \Leftrightarrow B$. Первое действие V_1 применяется к слову вида PAR и означает замену этого слова на слово PBR (если часть A отсутствует, то действие V_1 ничего не изменяет). Аналогично определяется действие второго вида.

В исчислении L_4 введены следующие соотношения слов:

$ek \Leftrightarrow ke$, $el \Leftrightarrow le$, $ea \Leftrightarrow ie$, $eb \Leftrightarrow jb$, $bm \Leftrightarrow lm$,
$hai \Leftrightarrow ah$, $hbj \Leftrightarrow bh$, $afeam \Leftrightarrow feam$,
$bfeam \Leftrightarrow feam$, $cfeam \Leftrightarrow feam$, $dfeam \Leftrightarrow feam$,
$af \Leftrightarrow fa$, $bf \Leftrightarrow fb$, $cf \Leftrightarrow fc$, $df \Leftrightarrow fd$, $ai \Leftrightarrow ia$,
$bi \Leftrightarrow ib$, $ci \Leftrightarrow ic$, $di \Leftrightarrow id$, $aj \Leftrightarrow ja$, $bj \Leftrightarrow jb$, $cj \Leftrightarrow jc$,
$dj \Leftrightarrow jd$, $ak \Leftrightarrow ka$, $bk \Leftrightarrow kb$, $ck \Leftrightarrow kc$, $dk \Leftrightarrow kd$,
$al \Leftrightarrow la$, $bl \Leftrightarrow lb$, $cl \Leftrightarrow lc$, $dl \Leftrightarrow ld$, $am \Leftrightarrow km$.

Здесь 37 соотношений, к ним добавлены 14 однобуквенных соотношений вида $\varphi \Leftrightarrow \varphi$, где $\varphi \in A_0$.

Допустимые действия относительно этих соотношений определяют правила вывода слов в L_4.

Определение 2. Слово Q *смежно со словом* P относительно L_4, если Q получается из P в результате одного допустимого действия, обозначение: $L_4 : P \vdash Q$. При этом соответствующее допустимое действие называется *правилом вывода* Q из P.

Следующие утверждения непосредственно вытекают из определения смежности.

Теорема 1. *Если* $L_4 : P \vdash Q$, *то* $L_4 : Q \vdash P$.

Теорема 2. *Если* $L_4 : P \vdash Q$, *то для любого слова* R *алфавита* A_0 $L_4 : RP \vdash RQ$ и $L_4 : PR \vdash QR$.

Определение 3. Пусть γ — буква, не входящая в A_0, и S является последовательностью слов, разделенных буквой γ вида $\gamma X_1 \gamma X_2 \gamma X_3 \gamma \ldots \gamma X_k \gamma$. Тогда S называется L_4-*рядом*, если всякий раз, когда слово X_i соседствует слева со словом X_{i+1}, то имеет место смежность $L_4 : X_i \vdash X_{i+1}$.

Определение 4. Слово Q называется *выводимым* из P в исчислении L_4, если можно построить L_4-ряд, соединяющий P с Q. Слова P и Q называются *эквивалентными* в L_4, если они взаимно выводимы в L_4, обозначение: $L_4 : P \Leftrightarrow Q$.

Следующие утверждения очевидны.

Теорема 3. *Если* $L_4 : P \vdash Q$, *то* $L_4 : P \Leftrightarrow Q$.

Теорема 4. *Если* $L_4 : P \Leftrightarrow Q$ *то* $L_4 : Q \Leftrightarrow P$.

Теорема 5. *Если* $L_4 : P \Leftrightarrow Q$, *то* $L_4 : RP \Leftrightarrow RQ$ *и* $L_4 : PR \Leftrightarrow QR$.

В [1] указаны другие ассоциативные исчисления, но главная особенность L_4 в следующем утверждении.

Теорема 6. *Множество слов, эквивалентных слову* $feam$ в L_4, *не является алгоритмически разрешимым.*

2. Базис автоматов. Описывается специальный базис C конечных автоматов из [4] и рассматриваются логические сети над C, предназначенные для моделирования следующей задачи.

Определение 5. Пусть P — слово алфавита A_0, и $\lambda(P)$ обозначает последовательность вида $\lambda(Q) = \mu\ldots\mu Q\mu\mu\ldots$, где длина приставки $\mu\ldots\mu$ равна $|Q|+1$, (здесь $|Q|$ обозначает длину слова Q). Тогда $\lambda(Q)$ называется *ограниченно-детерминированным оператором, определяемым словом* P.

Основная задача. Пусть P *и* Q — слова алфавита A_0, при этом Q является выводимым из P в L_4 с помощью последовательности правил вывода $V_1,\ldots,V_m$. Требуется построить логическую сеть K над базисом C, которая реализует оператор $\lambda(Q)$.

Автоматы базиса C разделяются на четыре типа *I-IV*. Наглядно они изображаются прямоугольниками

с входящей и выходящей стрелками, которые обозначают их входной и выходной каналы соответственно.

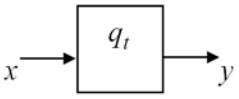

Рис. 1. Схема конечного автомата

Входные и выходные алфавиты всех автоматов фиксированы и состоят из символов множества $A = A_0 \cup A_0^1 \cup A_0^2 \cup A_1$, где $A_1 = \{\alpha, \beta, \delta, \mu, \mu^1, \mu^2, \mu^3, \omega\}$ и A_0^1, A_0^2 — множества букв алфавита A_0, помеченные верхними индексами 1 и 2 соответственно. Каждому автомату соответствует свое множество внутренних состояний. В любой момент времени t автомат находится в некотором внутреннем состоянии q_t, и на его входе расположен некоторый символ x из A. За один такт он переходит в некоторое состояние q_{t+1}, и на его выход поступает символ y. Такую команду можно изобразить следующим соотношением: $(q_t, x) \to (q_{t+1}, y)$. Программа автомата — это конечное множество команд указанного вида, которое не содержит различных команд с одинаковыми левыми частями. Программа может быть записана как множество буквенных векторов или в виде таблицы, в которой команды записываются в столбцах. В каждом автомате выделяется начальное состояние, и с него начинается работа автомата. В некоторые моменты на входах автомата могут отсутствовать какие-либо символы, тогда считается, что на них стоит пустой символ $\varnothing$, который учитывается при составлении программ.

Кроме того, каждый автомат может принимать специальное внутреннее состояние q_ω, которое называется *состоянием поломки* и удовлетворяет следующим требованиям:

1) в состоянии q_ω автомат поставляет на выход букву ω;

2) из состояния поломки он снова переходит в состояние q_ω;

3) если на вход автомата поступила буква, не принадлежащая A, то он переходит в состояние q_ω.

Определение 6. Говорят, что автомат *работает правильно*, если он не переходит в состояние поломки.

Базис C состоит из 117 автоматов, из них один автомат типа *I*, 88 автоматов типа *II* и по 14 автоматов типа *III* и *IV*. Программы этих автоматов представлены таблицами 1–4 в работе [4], а здесь приводится только описание основных моментов их работы.

Допустимая входная последовательность автомата типа *I* (при поступлении слева) имеет вид $\ldots\mu\mu a_n \ldots a_1\mu$. Символы этой последовательности поступают по тактам на вход автомата, он прибавляет к первым символам индекс 1 и поставляет на выход последовательность $\ldots\mu^2\mu^2\mu^1 a^1 \ldots a^1\mu^1$ Легко видеть, что если длина входной последовательности равна n+3, то автомат *I* выполняет n+4 такта.

Каждый автомат типа *II* соответствует некоторому правилу вывода исчисления L_4. В [4] для примера описана программа автомата II_{V_i}, моделирующего команду вида $AR \vdash RB$, где $A = r_1 \ldots r_m$, $B = s_1 \ldots s_n$, фиксированные непустые слова, и $R = p_1 \ldots p_k$ — произвольное слово. Если на вход II_{V_i} слева поступает последовательность $\mu^2 \ldots \mu^2\mu^1 p_k^1 \ldots p_1^1 r_m^1 \ldots r_1^1$, то он по тактам производит следующие действия:

1) заменяет символы $r_m^1 \ldots r_1^1$ символами $\mu^2 \ldots \mu^2\mu^2$;

2) затем поставляет слово $p_k^1 \ldots p_1^1\mu^1$;

3) присоединяет к нему $s_n^1 \ldots s_1^1$;

4) в конце выдает $\mu^2\mu^1$;

5) если поступает какая-то другая входная последовательность, то он принимает состояние поломки.

Легко видеть, что если пренебречь буквами μ и индексами остальных букв, то автомат II_{V_i} моделирует правило вывода V_i.

Каждый автомат типа *III* связан с некоторой буквой $z \in A_0$, и пусть III_z — обозначение такого автомата. В [4] показывается, что III_z подключается справа к автомату типа *II* или *III*. В первом случае если на его вход слева поступает последовательность вида $\mu^2\mu^1 a_m^1 \ldots a_1^1\mu^1\mu^2$, то III_z выполняет следующие действия:

1) поставляет на выход $\mu^3 z^2 \mu^3 \ldots \mu^3\mu^1$ до появления символа a_1^1;

2) если $a_1 = z$, то он поставляет символ μ^3 и буквы $a_2^1 \ldots a_m^1$ (если они имеются) до поступления μ^1;

3) при появлении μ^1 он выдает последовательность $\mu^1\mu^3\mu^3 \ldots$;

4) если $a_1 \neq z$, то он принимает состояние поломки.

Во втором случае на его вход поступает последовательность $\varnothing\mu^2\mu^2 \ldots \mu^2\mu^1 a_1^1 a_2^1 \ldots a_m^1\mu^1\mu^2 \ldots$ В этом случае III_z выполняет следующие действия:

1) поставляет на выход $\mu^3 z^2 \mu^3 \ldots \mu^3\mu^1$ до появления символа a_1^1;

2) если $a_1^1 = z^1$, то он поставляет на символ μ^3 и поступающие буквы $a_2^1 \ldots a_m^1$ (если они имеются) до появления μ^1;

3) при появлении μ^1 он выдает последовательность $\mu^1\mu^3\mu^3 \ldots$;

4) если $a_1^1 \neq z^1$, то он принимает состояние поломки. Аналогичные действия III_z выполняет во втором случае.

Программа каждого автомата типа *IV* также связана с некоторой буквой $z \in A_0$, такой автомат обозначается IV_z. В [4] показывается, что IV_z подключается справа к автомату типа *III* или типа *II*.

В первом случае на вход IV_z поступают символы четырех видов μ^3, μ^1, a^2 и a^1, где $a \in A_0$ У символов первых трех видов он удаляет индексы и поставляет их на выход. Далее, если первая поступившая на вход буква a^1 с индексом 1 совпадает с z^1, то IV_z

подает на выход свою букву z и принимает состояние q_1; а если $a^1 \neq z^1$, то он принимает состояние поломки. Если в состоянии q_1 на вход IV_z поступает символ μ^1, то он поставляет на выход последовательность $\mu\mu\ldots$ Аналогичные действия IV_z выполняет во втором случае.

3. Логические сети и реализуемые операторы. *Логической сетью над базисом C* называется ориентированный граф, в вершинах которого располагаются автоматы базиса *C*, а ребра графа изображают соединения входных и выходных каналов автоматов, входящих в сеть. Каждый выходной канал автомата подключается не более чем к одному входному каналу другого автомата и наоборот. В частности, исключаются простые петли, которые соединяют каналы одного и того же автомата. Естественным образом определяются *входной полюс и выходной полюс сети*, предполагается, что логическая сеть не содержит других свободных входных и выходных каналов автоматов, кроме полюсов. Для краткости ниже логические сети над C обозначаются буквой *K*.

Допустимой входной последовательностью сети K называется последовательность вида $\mu a_1 a_2 \ldots a_m \mu\mu\ldots$, где $a_1 a_2 \ldots a_n$ — непустое слово алфавита A_0. Ниже такие последовательности обозначаются буквой λ с некоторыми индексами.

Пусть $P = a_1 a_2 \ldots a_n,$, $Q = z_1 z_2 \ldots z_n$ — слова алфавита A_0, и Q выводимо из P в L_4 с помощью последовательности правил вывода $V_1,\ldots,V_m$. Следующая сеть K_0 называется *стандартным блоком автоматов, соответствующим паре* (*P, Q*) (рис. 2).

Все автоматы такого блока K_0 соединены последовательно. Первым стоит автомат типа *I*, за ним следуют автоматы типа *II*, далее идут автоматы типа *III*, и в конце один автомат типа *IV*. При поступлении допустимой входной последовательности λ_1 первый автомат сети K_0 выдает на выход некоторую последовательность λ_2. При правильной работе *I* последовательность λ_2 поступает на вход следующего автомата и т. д. Таким образом, если не возникнет состояния поломки, то на выходе сети K_0 появится некоторая последовательность λ_n, которая является *результатом работы* K_0.

В [4] доказываются следующие утверждения.

Теорема 7. *Логическая сеть* K_0 *реализует оператор вида*

$$\lambda(T) = \ldots \mu T \mu \ldots \mu.$$

Теорема 8. *Если в* L_4 *выводим отношение* $T \Leftrightarrow feam$, *то можно построить эффективно логическую сеть над C, реализующую оператор* $\lambda(T)$.

Теорема 9. Пусть *логическая сеть K над базисом C содержит только один автомат типа I и для некоторой входной последовательности* $\overline{x}$ *она работает правильно и вырабатывает последовательность* $\overline{y}$. *Тогда сушествуют слова P и Q, которые являются выводимыми в* L_4 *из слова feam, и для которых K является стандартным блоком автоматов, соответствующим паре* (*P, Q*)**.**

Теперь предлагается расширить виды логических сетей над базисом *C* следующим образом.

Пусть K_1 — стандартный блок базиса *C*, соответствующий (P_1, Q_1); и K_2 — аналогичный блок, соответствующий (P_2, Q_2). Выходной полюс K_1 присоединен слева к входному полюсу K_2. По определению блок K_1 для допустимой входной последовательности вырабатывает последовательность вида $\ldots\mu(z_k \ldots z_2 z_1)\mu\ldots\mu$. Такая последовательность может оказаться допустимой для блока K_2. В этом случае можно считать, что блок K_1 заканчивает свою работу и передает начальную часть своей последовательности блоку K_2 в качестве входной последовательности. И пусть автоматы блоков выбраны так, что объединенная сеть работает правильно. Это означает, что при согласованном подборе стандартные блоки можно последовательно соединять в логическую сеть, и такая сеть работает правильно.

Так как автоматы имеют только один вход и один выход, то других правильно организованных логических сетей не бывает.

Легко описываются условия, при которых такая сеть работает правильно.

Определение 7. Пусть *K* — правильно организованная сеть над базисом *C*. *Сложностью K н*азывается число автоматов типа *II,* входящих в эту сеть. Обозначение: *h (K*).

Теорема 10. Пусть S_n *обозначает множество операторов вида* $\lambda(T)$, *которые удовлетворяют условию* $h(K) \leq n$. *Тогда* S_n *является алгоритмически разрешимым.*

Доказательство (индукцией по *h (K*)). *Б*аза индукции очевидна. Далее, пусть множество S_n является алгоритмически разрешимым. Рассматриваем слово *T* алфавита A_0 и соответствующий $\lambda(T)$. Пусть *Q* — слово алфавита A_0, из которого слово *T* можно получить с помощью одного правила вывода исчисления L_4. В общем случае такое правило имеет вид: «замена *RAQ* на *PBQ*». При этом *T* = *RBQ* и соотношение $A \Leftrightarrow B$ является одним из известных соотношений исчисления L_4. Число таких слов *Q* конечно, и они находятся рекурсивно по *T*. Тогда если для некоторого такого *Q*

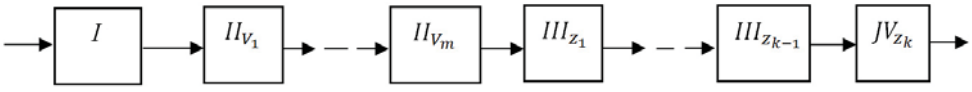

Рис. 2. Стандартный блок автоматов

выполняется $Q \in S_n$ то $h(T) \leq n + 1$, и в противном случае $h(T) > n + 1$. Теорема доказана.

В связи с доказанным утверждением возникает вопрос: как нам относиться к результату теоремы 6? Для данного слова T можно начать искать вывод T из слова *feam* путем перебора всех возможных выводов в L_4. И если искомый вывод короткий, то он будет найден. Таким образом, поставленная проблема сведена к поиску искомого объекта путем перебора всех возможных случаев. Но если искомый вывод очень длинный, то этот метод не поможет.

Библиографический список

1. Марков А.А., Нагорный Н.М. Теория алгорифмов. — М., 1984.

2. Кратко М.И. О существовании нерекурсивных базисов конечных автоматов // Алгебра и логика. — 1964. — Т. 3, №2.

3. Белякин Н.В. Усиление одной теоремы Мостовского // Вестник Сибирского Независимого института. — 2010. — №1.

4. Ганов В.А., Дегтерева Р.В. Алгоритмические проблемы конечных автоматов. — Барнаул, 2014.