ВЫЧИСЛИТЕЛЬНЫЕ МЕТОДЫ В ДИСКРЕТНОЙ МАТЕМАТИКЕ

УДК 519.726 DOI 10.17223/20710410/59/6

АЛГОРИТМ ВАЛИДАЦИИ ОГРАНИЧЕННО-ДЕТЕРМИНИРОВАННОГО ПОВЕДЕНИЯ ПЕРЕДАТЧИКА В КАНАЛЕ ЧАСТИЧНОГО СТИРАНИЯ

И. Б. Казаков

Московский физико-технический институт, г. Долгопрудный, Россия

E-mail: i_b_kazakov@mail.ru

Понятия структуры частичного стирания и канала частичного стирания были введены в предшествующих работах автора. Также в данных работах представлена формальная модель взаимодействия приемника и передатчика. Введено понятие корректного протокола, т. е. понятие согласования поведения приемника с поведением передатчика. Найдено накладываемое на поведение передатчика необходимое и достаточное условие того, что существует согласованное с ним поведение приемника. В настоящей работе представлен алгоритм проверки указанного условия и оценка его сложности.

Ключевые слова: *скрытые каналы, структура частичного стирания, канал частичного стирания, протокол передачи информации, алгоритм проверки.*

A VALIDATION ALGORITHM OF THE TRANSMITTER'S BOUNDEDLY DETERMINISTIC BEHAVIOUR IN A PARTIAL ERASURE CHANNEL

I. B. Kazakov

Moscow Institute of Physics and Technology, Dolgoprudny, Russia

In the previous papers, the notions of a partial erasure structure and a partial erasure channel have been introduced. A partial erasure structure is a triplet consisting of an alphabet A, a family of partitions of the alphabet and a set of probabilities assigned to the partitions. A partial erasure channel functions as follows. Alice sends Bob a symbol $a \in A$, but Bob receives only partial information about the symbol. Bob only knows which partition has been choosen and which class of the partition the symbol belongs to. The set of pairs consisting of a partition and a class of the partition, i.e., the signals Bob can receive, is denoted as Bob's alphabet B. There is a natural correspondence between the characters sent by Alice and received by Bob. For symbols $a \in A$ and $b \in B$, $a \mapsto b$ is assumed to hold if there exists a partition T in the partial erasure structure such that $b = (\pi_T(a), T)$. The correspondence is also generalized to the words. We assume that $a_1 \dots a_n \mapsto b_1 \dots b_n$ holds, if $a_i \mapsto b_i$ holds for all $i = 1 \dots n$. Suppose that Alice has an input tape from which she reads symbols of some finite alphabet S. Bob has an output tape, on which he can print symbols of alphabet S and also specially reserved (not in S) erasure symbol "*". Thus, a problem

of transmitting information via the descibed channel (i.e., a problem of construction of Alice's and Bob's behaviour) arises. In the previous papers of the author, a formal model of interaction between a transmitter (Alice) and a receiver (Bob), called a protocol, is presented. A protocol is a pair of functions (F, G), where $F : S^* \to A^*$ is the Alice behaviour function and $G : B^* \to S \cup \{*, \Lambda\}$ is the Bob behaviour function. Suppose that Alice has just read another symbol $s \in S$ from the input tape, having previously read the word $\hat{s} = s_1, \ldots, s_m$. Then we assume that during the following $|F(\hat{s}s)|$ steps Alice sends the word $F(\hat{s}s)$ symbol by symbol via the channel of partial erasure. Bob receives the symbol $b \in B$ on each step. After receiving the symbol, he has to decide what to print on the output tape. He can print a symbol of the alphabet S, or the erasure symbol "*", or nothing. Bob's decision is determined by the function G. The protocol (F, G) is said to be valid, if Bob prints on the output tape the same content what originally is on the input tape, while, perhaps, with the replacement of the significant symbols (i.e., the symbols of the alphabet S) to the erasure symbol "*". Among all Alice's behavior functions, a class of correct functions is defined. A function F is said to be correct, if $F(\Lambda)F(s_1^1)F(s_1^1 s_2^1)\ldots F(s_1^1\ldots s_n^1) \mapsto \beta_1$, $F(\Lambda)F(s_1^2)F(s_1^2 s_2^2)\ldots F(s_1^2\ldots s_m^2) \mapsto \beta_2$ and β_1 is a prefix to β_2, where $\beta_1, \beta_2 \in B^*$. The question is investigated, for which functions of Alice's behavior F there exists a function of Bob's behavior G such that the pair (F, G) is a correct protocol. The theorem is proved that for this it is necessary and sufficient that F belongs to the class of correct functions. This paper presents an algorithm for verifying that Alice's behavior function F belongs to the class of correct functions. The complexity of the algorithm is $O(L|Q|^3|S|^4)$, where $|Q|$ is the number of states of the automaton representing the function F, L is the maximum length of the word Alice throws out, $|S|$ is the number of symbols of the alphabet S.

Keywords: *covert channels, partial erasure structure, partial erasure channel, information transmission protocol, check algorithm.*

Введение

В работе представлен алгоритм, который по формальному описанию поведения передатчика (традиционно называемого Алисой) в канале частичного стирания проверяет выполнение критериального условия существования согласованного с ним поведения приемника (традиционно называемого Бобом). Понятие канала частичного стирания, формальное описание поведений Алисы и Боба, а также постановка задачи проверки указанного условия представлены в предшествующих работах автора [1–3].

Изложим сведения, обосновывающие актуальность решения поставленной задачи. Предмет, исследуемый в работе, имеет отношение к скрытым каналам. Скрытый канал — это любой коммуникационный канал, передающий информацию не предназначенным для этого способом. Понятие скрытого канала впервые в открытой литературе введено в работе [4]. Современный краткий обзор, посвященный скрытым каналам, можно найти, например, в [5]. Особую важность имеют сетевые скрытые каналы, обзор и классификация техник построения которых имеется в [6].

Настоящая работа связана со скрытым каналом блужданий по плоскости. Функционирование этого канала может быть описано следующим образом: передающий участник (Алиса) определённым образом движется по плоскости, а принимающий (Боб) считывает из наблюдаемых перемещений передающего участника закодированную им информацию. Скрытый канал блужданий по плоскости может быть реализован в online-шутерах, то есть в многопользовательских играх, где есть так называемое

«игровое поле», по которому возможны перемещения. Задача построения скрытых каналов через online-шутеры ранее исследовалась, например, в [7].

В работе [2] рассматривается предположение о том, что Боб получает лишь часть информации о движении Алисы. Имеется в виду, что он «видит» Алису, т. е. определяет её местоположение не во все моменты игрового времени, а только в некоторые из них. Таким образом, если две траектории Алисы совпадают в обозначенные «моменты видимости», то с точки зрения Боба данные траектории являются неразличимыми. Следовательно, на множестве траекторий Алисы, начинающихся в фиксированной точке и имеющих фиксированную длину N, определено 2^N отношений эквивалентности, каждое из которых соответствует определённому выбору множества «моментов видимости». Каждому из 2^N возможных выборов «моментов видимости» также приписана вероятность того, что Боб «увидит» Алису именно в выбранные моменты.

Далее произведено абстрагирование от траекторий и от множеств моментов видимости. В результате получены понятия структуры частичного стирания и канала частичного стирания.

Структура частичного стирания — это тройка, состоящая из алфавита A, семейства определённых на данном алфавите разбиений и набора вероятностей, приписанных разбиениям. Канал частичного стирания функционирует следующим образом. Алиса отправляет Бобу символ $a \in A$, а Боб получает только часть информации: какое было выбрано разбиение и какому классу выбранного разбиения принадлежал отправленный символ. Множество пар, состоящих из разбиения и класса по данному разбиению, т. е. множество возможных ответов, которые может получить Боб, обозначается как алфавит Боба B.

Легко видеть, что ранее упомянутое множество траекторий Алисы вместе с соответствующими отношениями эквивалентности является частным случаем структуры частичного стирания. Алфавитом A является множество траекторий. Каждое из 2^N отношений эквивалентности, заданных на траекториях, определяет разбиение алфавита, т. е. имеется множество разбиений, содержащее 2^N элементов. Каждому разбиению приписана соответствующая вероятность.

Таким образом, возникает задача передачи информации по описанному каналу, т. е. задача построения соответствующей пары поведений Алисы и Боба. Полагаем, что у Алисы имеется входная лента, с которой она читает символы некоторого конечного алфавита S. У Боба, в свою очередь, имеется выходная лента, на которую он может печатать символы алфавита S, а также специально зарезервированный (не входящий в S) символ стирания «*». Общей целью как Алисы, так и Боба является отпечатывание на выходной ленте того же содержания, которое изначально имеется на входной, при этом, может быть, с заменой значащих символов (т. е. символов алфавита S) на символ стирания «*». На множестве S определён порядок символов: $S = \{s_1, \ldots, s_d\}$. Также порядок определён и на множестве пар символов алфавита S, т. е. на множестве S^2. Данный порядок является лексикографическим, т. е. $(s_1, s_2) < (s'_1, s'_2)$, если $s_1 < s'_1$ или $s_1 = s'_1$ и $s_2 < s'_2$.

В работах [2, 3] рассмотрены формализованные схемы организации передачи информации от Алисы к Бобу, названные протоколами передачи информации в канале частичного стирания. В [2] описан частный случай протокола, называемый «схемой равномерного кодирования». Протоколу в общем случае посвящена работа [3]. В соответствии с произведённой в [3] формализацией поведение Алисы может быть описано как функция $F : S^* \to A^*$, а поведение Боба — как функция $G : B^* \to S \cup \{*\}$, где A^*, B^*, S^* — множества слов, т. е. конечных последовательностей символов ал-

фавитов A, B, S соответственно. Протокол — это пара функций (F, G). Функция F однозначно определяет детерминированную функцию $\hat{F}$ следующим образом: $\hat{F}(\hat{s}) = \hat{F}(s_1 \dots s_m) = F(\Lambda)F(s_1)F(s_1 s_2) \dots F(s_1 \dots s_m)$, где Λ — пустое слово.

Представим содержательную интерпретацию, относящуюся к функциям поведения F, G. Пусть Алиса только что прочитала с входной ленты очередной символ $s \in S$, предварительно уже считав слово $\hat{s} = s_1 \dots s_m$. Тогда считаем, что на протяжении последующих $|F(\hat{s}s)|$ тактов Алиса посимвольно отправляет по каналу частичного стирания слово $F(\hat{s}s)$. Что касается Боба, то на каждом такте он получает некий символ $b \in B$. Получив указанный символ, он должен принять решение о том, что печатать на выходной ленте: или какой-нибудь символ из $S \cup \{*\}$, или ничего не печатать. Полагаем, что если $G(\beta) \in S \cup \{*\}$, то Боб печатает символ $G(\beta)$. Иначе, т. е. если $G(\beta) = \Lambda$, Боб ничего не печатает.

В соответствии с общей целью Алисы и Боба в работе [3] введено понятие корректного протокола. Основным является вопрос о том, для каких именно функций поведения Алисы F существует функция поведения Боба G, такая, что пара (F, G) является корректным протоколом. Ответом является теорема, что для этого необходимо и достаточно, чтобы F принадлежала к классу правильных функций. Понятие правильной функции и доказательство теоремы представлены в [3]. Очевидно, представляет интерес решение задачи проверки произвольной функции $F : S^* \to A^*$ на принадлежность к классу правильных. Настоящая работа посвящена этому алгоритму и оценке его сложности.

Отметим, что функция $\hat{F}$ представима в виде (не обязательно конечного) автомата, у которого S является входным алфавитом, а множество слов A^* — выходным: автомат на каждом шаге принимает символ $s \in S$ и выдаёт слово $\alpha \in A^*$. Действительно, рассмотрим бесконечный автомат, состояния которого находятся во взаимно-однозначном соответствии с элементами множества S^*. Находясь в состоянии $\hat{s} \in S^*$ и приняв символ $s \in S$, автомат переходит в состояние $\hat{s}s$ и выдаёт слово $F(\hat{s}s)$. Подвергая указанный автомат преобразованию, отождествляющему его неразличимые состояния, получаем приведённый автомат, представляющий функцию $\hat{F}$.

Входными данными алгоритма считаются функции переходов и выходов описанного автомата. Алгоритм принимает входные данные и за конечное время выдаёт ответ вида «да/нет» о принадлежности функции поведения Алисы F к классу правильных.

Поскольку за конечное время алгоритм не может обработать бесконечный объём входных данных, то имеет смысл рассматривать только ограниченно-детерминированные функции, т. е. такие, автомат которых имеет конечное число состояний.

Зафиксируем функцию F, такую, что соответствующая ей функция $\hat{F}$ является ограниченно-детерминированной. Соответственно фиксированным окажется конечный автомат, представляющий функцию $\hat{F}$. Множество состояний автомата обозначим через Q. Автомат имеет функцию переходов $\varphi : Q \times S \to Q$ и выходную функцию $\psi : Q \times S \to A^*$. Начальное состояние обозначим q_Λ. В качестве $q_{\hat{s}}$ будем обозначать состояние, в которое автомат приходит, приняв на вход слово $\hat{s} = s_1 \dots s_m$, т. е. последовательно приняв символы $s_1, \dots, s_m$. Таким образом, для любого слова $\hat{s} \in S^*$ и любого символа $s \in S$ выполняется $\psi(q_{\hat{s}}, s) = F(\hat{s}s)$. Значение $F(\Lambda)$, отсутствующее в описании автомата, также фиксировано и хранится в памяти отдельно.

Представим структуру дальнейшего изложения. В разд. 1 изложено проверяемое условие правильности, в разд. 2 представлен алгоритм валидации и оценка его сложности. Доказательствам посвящён разд. 3. В заключении подведены итоги и намечены направления дальнейших исследований.

1. Условие правильности

Приведём формальное определение проверяемого условия. Необходимые вспомогательные понятия, а также определение правильной функции представлены в п. 1.1. В п. 1.2 данное определение преобразовано в равносильную форму, непосредственно проверяемую алгоритмом.

1.1. Определения

Повторим представленные в работах [2, 3] и упомянутые во введении понятия, относящиеся к функционированию канала частичного стирания. Для дальнейшего изложения также имеют значение пары слов из A^*, такие, что Боб всегда сможет отличить факт отправки Алисой одного слова из пары от факта отправки Алисой другого слова из этой пары.

Определение 1. Структура частичного стирания — тройка $(A, \mathfrak{T}, \mathfrak{P})$, состоящая из алфавита A, непустого множества определённых на нем разбиений (отношений эквивалентности) $\mathfrak{T}$, а также приписанных данным разбиениям весов p_T, сумма которых равна 1.

Определение 2. Два символа $a_1, a_2 \in A$ абсолютно различимы в структуре частичного стирания $(A, \mathfrak{T}, \mathfrak{P})$, если в ней не найдётся разбиения, такого, что a_1, a_2 принадлежат одному классу этого разбиения. Будем говорить, что слова α_1, α_2 имеют общую абсолютно различимую позицию в структуре частичного стирания $(A, \mathfrak{T}, \mathfrak{P})$, если найдётся такое i, что символы $\alpha_1[i], \alpha_2[i]$ абсолютно различимы в структуре частичного стирания $(A, \mathfrak{T}, \mathfrak{P})$.

Алиса отправляет Бобу символ $a \in A$. Боб получает лишь часть информации об отправленном символе: выбранное отношение эквивалентности T, а также класс $\pi_T(a)$ по данному отношению. Множество всех пар $(\pi_T(a), T)$ обозначается как алфавит B Боба. Таким образом, между отправляемыми Алисой и получаемыми Бобом символами естественным образом определено вероятностное соответствие $a \underset{p_T}{\mapsto} b$. Поскольку в настоящей работе не рассматриваются вероятностные аспекты исследуемой модели, будем использовать обозначение $a \mapsto b$. Данное соответствие обобщается также на слова.

Определение 3. Для символов $a \in A$, $b \in B$ полагаем $a \mapsto b$, если в структуре частичного стирания имеется разбиение T, такое, что $b = (\pi_T(a), T)$. Для слов $\alpha \in A^*$, $\beta \in B^*$ полагаем $\alpha \mapsto \beta$, если $|\alpha| = |\beta| = l$ и для всех $i = 1, \ldots, l$ выполнено $\alpha(i) \mapsto \beta(i)$.

Определение 4. Для функции F определим функцию $\hat{F} : S^* \to A^*$ по следующему правилу: $\hat{F}(\hat{s}) = F(\Lambda)F(s_1)F(s_1 s_2)\ldots F(s_1 \ldots s_m)$, где $\hat{s} = s_1 \ldots s_m$.

Определение 5. Слово β_1 — префикс слова β_2, если β_2 имеет вид $\beta_2 = \beta_1\beta_1'$. Аналогично, β_1 — суффикс слова β_2, если β_2 имеет вид $\beta_2 = \beta_1'\beta_1$.

Определение 6. Функция F называется правильной, если из выполнения условий $\hat{F}(\hat{s}_1) \mapsto \beta_1$, $\hat{F}(\hat{s}_2) \mapsto \beta_2$ и β_1 — префикс β_2 следует выполнение условия $|\hat{s}_1| \leqslant |\hat{s}_2|$.

1.2. Равносильный вид

Преобразуем условие правильности таким образом, чтобы входящие в его формулировку понятия относились только к структуре частичного стирания и к функции поведения Алисы F. Для этого введём вспомогательные понятия аномальных пар слов из S^* 1-го и 2-го рода.

Определение 7. Пара слов $(\hat{s}_1, \hat{s}_2) \in (S^*)^2$ называется одноходовой, если выполнено $|\hat{s}_1| = |\hat{s}_2| + 1$.

Определение 8. Пара слов $(\hat{s}_1, \hat{s}_2) \in (S^*)^2$ называется аномалией 1-го рода для функции F, если слова $\hat{F}(\hat{s}_1)$ и $\hat{F}(\hat{s}_2)$ имеют общую абсолютно различимую позицию.

Определение 9. Пара слов $(\hat{s}_1, \hat{s}_2) \in (S^*)^2$ называется аномалией 2-го рода для функции F, если $(|\hat{F}(\hat{s}_1)| \leqslant |\hat{F}(\hat{s}_2)|$ и $|\hat{s}_2| > |\hat{s}_1|)$ или $(|\hat{F}(\hat{s}_2)| \leqslant |\hat{F}(\hat{s}_1)|$ и $|\hat{s}_1| < |\hat{s}_2|)$.

Теорема 1. Для того чтобы функция F была правильной, необходимо и достаточно несуществования одноходовой пары, которая является аномалией 2-го рода, но не является аномалией 1-го рода для функции F.

Доказательство.

Необходимость. Предположим обратное: $(\hat{s}_1, \hat{s}_2)$ — аномалия 2-го рода для функции F, не являющаяся аномалией 1-го рода для функции F, т. е. $|\hat{s}_1| = |\hat{s}_2| + 1$, $|\hat{F}(\hat{s}_1)| \leqslant |\hat{F}(\hat{s}_2)|$, а слова $\hat{F}(\hat{s}_1), \hat{F}(\hat{s}_2)$ не имеют общей абсолютно различимой позиции.

Обозначим: $\alpha_1 = \hat{F}(\hat{s}_1)$, $\alpha_2 = \hat{F}(\hat{s}_2)$, $k_1 = |\alpha_1|$, $k_2 = |\alpha_2|$. Слова α_1, α_2 не имеют общей абсолютно различимой позиции и $k_1 \leqslant k_2$.

Так как слова α_1, α_2 не имеют общей абсолютно различимой позиции, можно выбрать k_1 разбиений $T_1, \ldots, T_{k_1}$ из структуры частичного стирания, таких, что при всех $i = 1, \ldots, k_1$ символы $\alpha_1(i), \alpha_2(i)$ принадлежат одному классу разбиения T_i, т. е. выполняется $\pi_{T_i}(\alpha_1(i)) = \pi_{T_i}(\alpha_2(i))$. Определим $b_i = (T_i, \pi_{T_i}(\alpha_1(i)))$; таким образом, при всех $i = 1, \ldots, k_1$ выполнено $\alpha_1(i), \alpha_2(i) \mapsto b_i$.

Выберем произвольным образом оставшиеся $k_2 - k_1$ разбиений $T_{k_1+1}, \ldots, T_{k_2}$. Определим $b_i = (T_i, \pi_{T_i}(\alpha_2(i)))$ при $i = k_1 + 1, \ldots, k_2$. Таким образом, при $i = k_1 + 1, \ldots, k_2$ выполняется $\alpha_2(i) \mapsto b_i$.

Положим $\beta_1 = b_1 \ldots b_{k_1}$, $\beta_2 = b_1 \ldots b_{k_1} b_{k_1+1} \ldots b_{k_2}$. Тогда $\hat{F}(\hat{s}_1) = \alpha_1 \mapsto \beta_1$, $\hat{F}(\hat{s}_2) = \alpha_2 \mapsto \beta_2$, β_1 — префикс β_2. Согласно определению правильной функции, отсюда следует, что выполняется $|\hat{s}_2| + 1 = |\hat{s}_1| \leqslant |\hat{s}_2|$. Противоречие.

Достаточность.

1. Предположим обратное: F не является правильной функцией. Это означает существование слов $\hat{s}_1, \hat{s}_2 \in S^*$ и $\beta_1, \beta_2 \in B^*$, таких, что β_1 — префикс β_2, $\hat{F}(\hat{s}_1) \mapsto \beta_1$, $\hat{F}(\hat{s}_2) \mapsto \beta_2$, $|\hat{s}_1| > |\hat{s}_2|$.

2. Отсюда следует, что слова $\hat{F}(\hat{s}_1)$ и $\hat{F}(\hat{s}_2)$ не имеют общей абсолютно различимой позиции, т. е. пара $(\hat{s}_1, \hat{s}_2)$ не является аномалией 1-го рода для функции F.

3. Поскольку $|\hat{s}_1| > |\hat{s}_2|$, то можно записать $\hat{s}_1 = \hat{s}'_1 \hat{s}''_1$, где $|\hat{s}'_1| = |\hat{s}_2| + 1$. Отметим, что так как $\hat{F}(\hat{s}'_1)$ — префикс $\hat{F}(\hat{s}_1)$, то одноходовая пара $(\hat{s}'_1, \hat{s}_2)$ также не является аномалией 1-го рода для функции F.

4. Так как $\hat{s}'_1$ — префикс $\hat{s}_1$, то $\hat{F}(\hat{s}'_1)$ — префикс $\hat{F}(\hat{s}_1)$.

5. Определим слово β'_1 как префикс слова β_1, такой, что $|\beta'_1| = |\hat{F}(\hat{s}'_1)|$. Очевидно, что $\hat{F}(\hat{s}'_1) \mapsto \beta'_1$.

6. Итак, β'_1 — префикс β_1 (п. 1), β_1 — префикс β_2 (п. 5), следовательно, β'_1 — префикс β_2.

7. Таким образом, $|\hat{F}(\hat{s}'_1)| = |\beta'_1| \leqslant |\beta_2| = |\hat{F}(\hat{s}_2)|$. Следовательно, одноходовая пара $(\hat{s}'_1, \hat{s}_2)$ является аномалией 2-го рода для функции F.

8. Сопоставляя выводы п. 3 и 7, получаем противоречие.

Теорема 1 доказана. ■

Указанное в теореме 1 равносильное условие непосредственно проверяется алгоритмом.

2. Алгоритм валидации

Входными данными алгоритма являются функция перехода φ и выходная функция ψ конечного автомата, представляющего функцию $\hat{F}$. Функции заданы в виде таблиц. Отдельно в памяти хранится слово $F(\Lambda)$, хотя фактически оно не используется алгоритмом.

Алгоритм состоит из подготовительной и основной части. Описание подготовительной части алгоритма, называемого процедурой предынициализации, представлено в п. 2.1, основной части — в п. 2.3. Основная часть алгоритма работает с так называемыми перебираемыми сущностями, которые предварительно вводятся в п. 2.2.

2.1. Процедура предынициализации

До основной работы алгоритма выполняется процедура, модифицирующая входные данные, т. е. изменяющая функцию F таким образом, чтобы во время основной работы оказались выполнены ограничения $F(\Lambda) = \Lambda$ и $F(\hat{s}) \neq \Lambda$ при всех $\hat{s} \neq \Lambda$.

В первую очередь хранящееся в памяти значение $F(\Lambda)$ заменяется на пустое слово Λ, то есть функция F заменяется на функцию F_0, определяемую следующим образом: $F_0(\Lambda) = \Lambda$; $F_0(\hat{s}) = F(\hat{s})$, если $\hat{s} \neq \Lambda$. Эта замена корректна, так как F_0 правильна тогда и только тогда, когда правильна F. Данная операция требует времени $O(1)$.

Во вторую очередь проверяем, выполнено ли для всех слов $\hat{s} \neq \Lambda$ условие $F(\hat{s}) \neq \Lambda$, т. е. $\psi(q, s) \neq \Lambda$ для всех состояний $q \in Q$ представляющего автомата и всех символов $s \in S$. Указанная проверка выполняется за время, пропорциональное времени перечисления всех возможных переходов между состояниями, т. е. за время $O(|Q||S|)$. Если условие не выполнено, то функция F не является правильной и алгоритм возвращает ответ «нет» и завершается. Если условие выполнено, то алгоритм переходит к выполнению основной работы.

Теорема 2. Пусть функция F_0 определена следующим образом: $F_0(\Lambda) = \Lambda$; $F_0(\hat{s}) = F(\hat{s})$, если $\hat{s} \neq \Lambda$. Тогда функция F_0 правильна тогда и только тогда, когда правильна функция F.

Доказательство. Зафиксируем пару слов $(\hat{s}_1, \hat{s}_2) \in (S^*)^2$. Слова $\hat{F}_0(\hat{s}_1)$ и $\hat{F}_0(\hat{s}_2)$ имеют общую абсолютно различимую позицию тогда и только тогда, когда её имеют слова $\hat{F}(\hat{s}_1) = F(\Lambda)\hat{F}_0(\hat{s}_1)$, $\hat{F}(\hat{s}_2) = F(\Lambda)\hat{F}_0(\hat{s}_2)$. Равносильны следующие условия: $|\hat{F}(\hat{s}_1)| \leqslant |\hat{F}(\hat{s}_2)|$ и $|\hat{F}_0(\hat{s}_1)| \leqslant |\hat{F}_0(\hat{s}_2)|$, поскольку $|\hat{F}(\hat{s}_1)| = |\hat{F}_0(\hat{s}_1)| + |F(\Lambda)|$ и $|\hat{F}(\hat{s}_1)| = |\hat{F}_0(\hat{s}_1)| + |F(\Lambda)|$

Таким образом, пара слов $(\hat{s}_1, \hat{s}_2)$ является аномалией 2-го рода, но не является аномалией 1-го рода для функции F_0 в том и только в том случае, когда она является таковой для функции F. Применяя теорему 1, получаем требуемое. ■

Теорема 3. Пусть для некоторого слова $\hat{s} \in S^*$ выполнено $\hat{s} \neq \Lambda$ и $\hat{F}(\hat{s}) = \Lambda$. Тогда функция F не является правильной.

Доказательство. Предположим обратное: пусть F — правильная функция.

Так как $\hat{s} \neq \Lambda$, то существуют слово $\hat{s}' \in S^*$ и символ $s \in S$, такие, что $\hat{s} = \hat{s}'s$.

Условие теоремы означает, что $\alpha = \hat{F}(\hat{s}) = \hat{F}(\hat{s}'s) = \hat{F}(\hat{s}')F(\hat{s}'s) = \hat{F}(\hat{s}')F(\hat{s}) = \hat{F}(\hat{s}')\Lambda = \hat{F}(\hat{s}')$.

Обозначим $k = |\alpha|$. Выберем из структуры частичного стирания k произвольных разбиений $T_1, \ldots, T_k$ и положим $b_i = (T_i, \pi_{T_i}(\alpha(i)))$, $\beta = b_1 \ldots b_k$. Тогда $\alpha(i) \mapsto b_i$ и, следовательно, $\hat{F}(\hat{s}'s) = \hat{F}(\hat{s}') = \alpha \mapsto \beta$.

Таким образом, $\hat{F}(\hat{s}'s), \hat{F}(\hat{s}') \mapsto \beta$. Так как β — префикс самого себя, то, применяя определение правильной функции, получаем $|\hat{s}'| + 1 = |\hat{s}'s| \leqslant |\hat{s}'|$. Противоречие. ■

2.2. Перебираемые сущности

Алгоритм оперирует с пятёрками вида $(q_1, q_2, \alpha, \hat{s}_1, \hat{s}_2)$, состоящими из двух состояний $q_1, q_2 \in Q$ представляющего функцию $\hat{F}$ автомата, слова α из множества A^*, а также двух слов $\hat{s}_1, \hat{s}_2$ из множества S^*. Такая пятёрка называется перебираемой сущностью.

Определение 10. Перебираемая сущность — это пятёрка $(q_1, q_2, \alpha, \hat{s}_1, \hat{s}_2)$, где $q_1, q_2 \in Q$; $\alpha \in A^*$; $\hat{s}_1, \hat{s}_2 \in S^*$.

Определение 11. Будем говорить, что перебираемая сущность $(q_1, q_2, \alpha, \hat{s}_1, \hat{s}_2)$ имеет размер k, если $|\hat{s}_2| = k$. Размер перебираемой сущности e будем обозначать как $\mathrm{size}(e)$.

На множестве перебираемых сущностей определено действие пар символов (s_1, s_2) из S. Под действием указанной пары перебираемая сущность может переходить в аномалию 1-го рода, или в аномалию 2-го рода, или в другую перебираемую сущность.

Определение 12. Перебираемая сущность $(q_1, q_2, \alpha, \hat{s}_1, \hat{s}_2)$ под действием пары символов (s_1, s_2) переходит в аномалию 1-го рода, если слова $\alpha\psi(q_1, s_1)$ и $\psi(q_2, s_2)$ имеют общую абсолютно различимую позицию.

Определение 13. Перебираемая сущность $(q_1, q_2, \alpha, \hat{s}_1, \hat{s}_2)$ под действием пары символов (s_1, s_2) переходит в аномалию 2-го рода, если $|\alpha\psi(q_1, s_1)| \leqslant |\psi(q_2, s_2)|$.

Определение 14. Перебираемая сущность $(q_1, q_2, \alpha, \hat{s}_1, \hat{s}_2)$ под действием пары символов (s_1, s_2) переходит в перебираемую сущность $(q_1', q_2', \alpha', \hat{s}_1', \hat{s}_2')$, если она под действием этой пары не переходит ни в аномалию 1-го рода, ни в аномалию 2-го рода, а также выполнены следующие условия: $q_1' = \varphi(q_1, s_1)$, $q_2' = \varphi(q_2, s_2)$, $\hat{s}_1' = \hat{s}_1 s_1$, $\hat{s}_2' = \hat{s}_2 s_2$, α' — суффикс $\alpha\psi(q_1, s_1)$, $|\alpha'| = |\alpha\psi(q_1, s_1)| - |\psi(q_2, s_2)|$.

Определение 15. Будем говорить, что перебираемые сущности $(q_1, q_2, \alpha, \hat{s}_1, \hat{s}_2)$ и $(q_1', q_2', \alpha', \hat{s}_1', \hat{s}_2')$ подобны, если $q_1 = q_1'$, $q_2 = q_2'$ и $\alpha = \alpha'$.

Установим свойства подобных сущностей:

Утверждение 1. Пусть под действием пары символов (s_1, s_2) перебираемые сущности e_1, e_2 переходят в перебираемые сущности e_1', e_2' соответственно. Если сущности e_1, e_2 подобны, то подобны и сущности e_1', e_2'

Доказательство. Сущности e_1, e_2 подобны, то есть $e_1 = (q_1, q_2, \alpha, \hat{s}_{11}, \hat{s}_{12})$, $e_2 = (q_1, q_2, \alpha, \hat{s}_{21}, \hat{s}_{22})$. Под действием пары символов (s_1, s_2) сущности e_1, e_2 переходят в сущности $e_1' = (\varphi(q_1, s_1), \varphi(q_2, s_2), \alpha', \hat{s}_{11}s_1, \hat{s}_{12}s_2)$, $e_2' = (\varphi(q_1, s_1), \varphi(q_2, s_2), \alpha', \hat{s}_{11}s_1, \hat{s}_{12}s_2)$ соответственно, где α' — суффикс $\alpha\psi(q_1, s_1)$, и выполнено $|\alpha'| = |\alpha\psi(q_1, s_1)| - |\psi(q_2, s_2)|$. У сущностей e_1', e_2' совпадают первые три компоненты, т. е. они подобны. ■

Утверждение 2. Пусть e_1, e_2 — подобные перебираемые сущности, (s_1, s_2) — пара символов алфавита S. Тогда

1) Если под действием (s_1, s_2) сущность e_1 переходит в аномалию 1-го рода, то e_2 также переходит в аномалию 1-го рода.
2) Если под действием (s_1, s_2) сущность e_1 переходит в аномалию 2-го рода, то e_2 также переходит а аномалию 2-го рода.
3) Если под действием (s_1, s_2) сущность e_1 переходит в некую сущность e_1', то найдется сущность e_2', такая, что e_2 переходит в e_2' под действием (s_1, s_2).

Доказательство. Сущности e_1, e_2 подобны, то есть $e_1 = (q_1, q_2, \alpha, \hat{s}_{11}, \hat{s}_{12})$, $e_2 = (q_1, q_2, \alpha, \hat{s}_{21}, \hat{s}_{22})$.

1) Если под действием (s_1, s_2) сущность e_1 переходит в аномалию 1-го рода, то слова $\alpha\psi(q_1, s_1)$ и $\psi(q_2, s_2)$ имеют общую абсолютно различимую позицию. Тогда сущность e_2 также переходит в аномалию 1-го рода под действием (s_1, s_2).

2) Если под действием (s_1, s_2) сущность e_1 переходит в аномалию 2-го рода, то $|\alpha\psi(q_1, s_1)| \leqslant |\psi(q_2, s_2)|$. Тогда сущность e_2 также переходит в аномалию 2-го рода под действием (s_1, s_2).

3) Если под действием (s_1, s_2) сущность e_1 переходит в некую сущность e_1', то e_1 не переходит ни в аномалию 1-го рода, ни в аномалию 2-го рода под действием (s_1, s_2). Тогда слова $\alpha\psi(q_1, s_1)$ и $\psi(q_2, s_2)$ не имеют общей абсолютно различимой позиции и $|\alpha\psi(q_1, s_1)| > |\psi(q_2, s_2)|$. Следовательно, e_2 также не переходит ни в аномалию 1-го рода, ни в аномалию 2-го рода. Таким образом, найдётся сущность e_2', такая, что e_2 переходит в e_2' под действием (s_1, s_2).

Утверждение 2 доказано. ■

2.3. Описание основной работы алгоритма

Кроме входных данных, алгоритм хранит в памяти, во-первых, очередь перебираемых сущностей, которые предстоит обработать алгоритму. Во-вторых, в памяти хранится также множество перебираемых сущностей, которое далее называется множеством принятых к обработке. Тот факт, что конкретная перебираемая сущность в некоторый момент находится в этом множестве, означает, что к данному моменту алгоритм уже добавлял в очередь подобную ей перебираемую сущность.

Вызываемые алгоритмом 1 процедуры размещают в очереди перебираемые сущности. Разместить в очереди перебираемую сущность $(q_1, q_2, \alpha, \hat{s}_1, \hat{s}_2)$ означает предварительно проверить, не находится ли подобная ей перебираемая сущность в множестве принятых к обработке. Если не находится, то сущность $(q_1, q_2, \alpha, \hat{s}_1, \hat{s}_2)$ следует добавить в хвост очереди и в множество принятых к обработке. В ином случае ничего не следует делать.

Алгоритм 1. Основная часть алгоритма валидации

1: Выполнить процедуру инициализации:
2: **Для всех** символов $s \in S$, взятых в соответствующем порядке,
разместить в очереди инициальную перебираемую сущность $(q_s, q_\Lambda, \hat{F}(s), s, \Lambda)$.
3: **Пока** очередь не пуста и не выдано исключение:
брать с головы очереди перебираемую сущность $(q_1, q_2, \alpha, \hat{s}_1, \hat{s}_2)$ и подавать её как аргумент на вход процедуры обработки:
4: **Для всех** пар символов $(s_1, s_2) \in S^2$, взятых в лексикографическом порядке:
5: **Если** перебираемая сущность $(q_1, q_2, \alpha, \hat{s}_1, \hat{s}_2)$ под действием пары (s_1, s_2) переходит в аномалию 1-го рода, **то**
6: пропустить выполнение последующих шагов цикла.
7: **Если** перебираемая сущность $(q_1, q_2, \alpha, \hat{s}_1, \hat{s}_2)$ под действием пары (s_1, s_2) переходит в аномалию 2-го рода, **то**
8: выдать исключение, т. е. завершить работу алгоритма и вернуть ответ «нет».
9: Для перебираемой сущности $(q_1, q_2, \alpha, \hat{s}_1, \hat{s}_2)$ найти, в какую другую перебираемую сущность $(q_1', q_2', \alpha', \hat{s}_1', \hat{s}_2')$ она переходит под действием пары символов (s_1, s_2). Такая сущность найдётся, так как не осуществляется перехода ни в аномалию 1-го, ни в аномалию 2-го рода.
10: Разместить в очереди перебираемую сущность $(q_1', q_2', \alpha', \hat{s}_1', \hat{s}_2')$.

Определение 16. Перебираемая сущность называется инициальной, если она имеет вид $(q_s, q_\Lambda, \hat{F}(s), s, \Lambda)$, где символ $s \in S$ рассматривается как слово длины 1.

Формально для алгоритма имеются три возможности: или он остановится в состоянии пустой очереди, или выдаст исключение, или будет работать бесконечно долго. Представляющий функцию F автомат имеет $|S|$ входных символов и $|Q|$ внутренних состояний. Значениями выходной функции являются слова из A^*. Обозначим максимальную длину слова из множества значений выходной функции через L.

Теорема 4. Возможность бесконечно долгой работы алгоритма исключена. Если F — правильная функция, то алгоритм остановится в состоянии пустой очереди, иначе он выдаст исключение. Сложность алгоритма составляет $O(L|Q|^3|S|^4)$.

3. Доказательство теоремы 4

В п. 3.1 определим вид полных перебираемых сущностей, которые могут быть поданы как аргумент в процедуру обработки. Далее в п. 3.2 установим такие свойства алгоритма, как «единственность обработки краткой сущности» и «упорядоченность по размеру». Условия выдачи алгоритмом исключения изучены в п. 3.3 и применены в п. 3.4 к аномалиям, в результате чего установлено, что исключение выдаётся тогда и только тогда, когда проверяемая функция не является правильной. В п. 3.5 показывается, что некоторые сущности не могут обрабатываться алгоритмом, тем самым обосновывается конечность времени его работы. Оценка сложности представлена в п. 3.6.

Отметим, что поскольку алгоритм вызывает процедуру предынициализации, то считаем, что на функцию F наложены соответствующие ограничения: $F(\Lambda) = \Lambda$, $F(\hat{s}) \neq \Lambda$ для всех $\hat{s} \neq \Lambda$.

3.1. Регулярные сущности

Определим вид перебираемых сущностей, которые могут быть поданы как аргумент в функцию обработки. Такие сущности назовём регулярными.

Определение 17. Будем говорить, что одноходовая пара слов $(\hat{s}_1, \hat{s}_2)$ из S^* не содержит префиксным образом аномалию 2-го рода для F, если никакая одноходовая пара $(\hat{s}'_1, \hat{s}'_2)$, такая, что $\hat{s}'_i$ — префикс $\hat{s}_i$, не является аномалией 2-го рода для F.

Определение 18. Перебираемая сущность $(q_1, q_2, \alpha, \hat{s}_1, \hat{s}_2)$ называется регулярной, если выполнены следующие условия: $q_1 = q_{\hat{s}_1}$; $q_2 = q_{\hat{s}_2}$; $(\hat{s}_1, \hat{s}_2)$ — одноходовая пара, которая не является аномалией 1-го рода для функции F и не содержит префиксным образом аномалию 2-го рода для функции F; слово α — суффикс слова $\hat{F}(\hat{s}_1)$; $|\alpha| = |\hat{F}(\hat{s}_1)| - |\hat{F}(\hat{s}_2)|$.

Легко видеть, что понятия «инициальная сущность» и «регулярная сущность размера 0» тождественны.

Установим свойства регулярных сущностей, гарантирующие, что алгоритмом будут обрабатываться только регулярные сущности. Во-первых, инициальные сущности регулярны. Во-вторых, регулярная сущность, если не переходит в аномалию 1-го или 2-го рода, под действием любой пары символов (s_1, s_2) переходит в регулярную сущность. В-третьих, всякая неинициальная регулярная сущность имеет предшествующую регулярную сущность, т. е. ту, которая под действием некоторой пары (s_1, s_2) переходит в данную неинициальную регулярную сущность.

Утверждение 3. Пусть слова $\hat{s}'_1, \hat{s}'_2$ — префиксы слов $\hat{s}_1, \hat{s}_2$ соответственно и пара $(\hat{s}'_1, \hat{s}'_2)$ является аномалией 1-го рода для функции F. Тогда пара $(\hat{s}_1, \hat{s}_2)$ также является аномалией 1-го рода для функции F.

Доказательство. Так как $\hat{s}'_1$ — префикс $\hat{s}_1$, то $\hat{F}(\hat{s}'_1)$ — префикс $\hat{F}(\hat{s}_1)$. Аналогично, из того, что $\hat{s}'_2$ — префикс $\hat{s}_2$, следует, что $\hat{F}(\hat{s}'_2)$ — префикс $\hat{F}(\hat{s}_2)$.

Пусть слова $\hat{F}(\hat{s}'_1)$ и $\hat{F}(\hat{s}'_2)$ имеют общую абсолютно различимую позицию. Тогда данная позиция является также абсолютно различимой позицией слов $\hat{F}(\hat{s}'_1)$ и $\hat{F}(\hat{s}'_2)$. Следовательно, пара $(\hat{s}_1, \hat{s}_2)$ — аномалия 1-го рода для функции F. ■

Утверждение 4. Пусть слова $\hat{s}'_1, \hat{s}'_2$ — префиксы слов $\hat{s}_1, \hat{s}_2$ соответственно, пары слов $(\hat{s}'_1, \hat{s}'_2)$ и $(\hat{s}_1, \hat{s}_2)$ являются одноходовыми и пара $(\hat{s}_1, \hat{s}_2)$ не содержит префиксным образом аномалию 2-го рода для функции F. Тогда пара $(\hat{s}'_1, \hat{s}'_2)$ также не содержит префиксным образом аномалию 2-го рода для функции F.

Доказательство. Предположим обратное, т. е. что $(\hat{s}'_1, \hat{s}'_2)$ содержит префиксным образом аномалию 2-го рода для функции F. Тогда найдётся одноходовая пара слов $(\hat{s}''_1, \hat{s}''_2)$, являющаяся аномалией 2-го рода для функции F, такая, что $\hat{s}''_1$ — префикс $\hat{s}'_1$, $\hat{s}''_2$ — префикс $\hat{s}'_2$. Так как, в свою очередь, $\hat{s}'_1$ — префикс $\hat{s}_1$, то $\hat{s}''_1$ — префикс $\hat{s}_1$; аналогично, $\hat{s}''_2$ — префикс $\hat{s}_2$.

Из свойств пары $(\hat{s}''_1, \hat{s}''_2)$ следует, что пара $(\hat{s}_1, \hat{s}_2)$ содержит префиксным образом аномалию 2-го рода для функции F, что противоречит условию. ■

Утверждение 5. Пусть $(\hat{s}_1, \hat{s}_2)$ — одноходовая пара слов алфавита S, s_1, s_2 — символы алфавита S. Если пара $(\hat{s}_1 s_1, \hat{s}_2 s_2)$ не является аномалией 2-го рода для функции F, а пара $(\hat{s}_1, \hat{s}_2)$ не содержит префиксным образом аномалию 2-го рода для функции F, то пара $(\hat{s}_1 s_1, \hat{s}_2 s_2)$ также не содержит префиксным образом аномалию 2-го рода для функции F.

Доказательство. Предположим обратное, т. е. что пара $(\hat{s}_1 s_1, \hat{s}_2 s_2)$ содержит префиксным образом аномалию 2-го рода для функции F. Тогда найдётся одноходовая пара слов $(\hat{s}'_1, \hat{s}'_2)$, являющаяся аномалией 2-го рода для функции F, такая, что $\hat{s}'_1$ — префикс $\hat{s}_1 s_1$, а $\hat{s}'_2$ — префикс $\hat{s}_2 s_2$.

Предположим, что $\hat{s}'_2 = \hat{s}_2 s_2$. Тогда $|\hat{s}'_1| = |\hat{s}'_2| + 1 = |\hat{s}_2 s_2| + 1 = |\hat{s}_2| + 2 = |\hat{s}_1| + 1 = |\hat{s}_1 s_1|$. Так как $\hat{s}'_1$ — префикс $\hat{s}_1 s_1$, то отсюда следует $\hat{s}'_1 = \hat{s}_1 s_1$. Следовательно, пара $(\hat{s}_1 s_1, \hat{s}_2 s_2)$ является аномалией 2-го рода для функции F, что противоречит условию. Таким образом, $\hat{s}'_2 \neq \hat{s}_2 s_2$.

Так как $\hat{s}'_2$ — префикс $\hat{s}_2 s_2$ и $\hat{s}'_2 \neq \hat{s}_2 s_2$, то $|\hat{s}'_2| < |\hat{s}_2 s_2|$. Отсюда следует $|\hat{s}'_1| = |\hat{s}'_2| + 1 < |\hat{s}_2 s_2| + 1 = |\hat{s}_2| + 2 = |\hat{s}_1| + 1 = |\hat{s}_1 s_1|$. Таким образом, $\hat{s}'_1 \neq \hat{s}_1 s_1$.

Так как $\hat{s}'_1$ — префикс $\hat{s}_1 s_1$ и $\hat{s}'_1 \neq \hat{s}_1 s_1$, то $\hat{s}'_1$ — префикс $\hat{s}_1$.

Так как $\hat{s}'_2$ — префикс $\hat{s}_2 s_2$ и $\hat{s}'_2 \neq \hat{s}_2 s_2$, то $\hat{s}'_2$ — префикс $\hat{s}_2$.

Из свойств пары $(\hat{s}'_1, \hat{s}'_2)$ следует, что пара $(\hat{s}_1, \hat{s}_2)$ содержит префиксным образом аномалию 2-го рода для функции F, что противоречит условию. ■

Утверждение 6. Инициальные перебираемые сущности, т. е. сущности вида $(q_s, q_\Lambda, F(s), s, \Lambda)$, $s \in S$, являются регулярными.

Доказательство.

1. Пара вида (s, Λ) не может быть аномалией 1-го рода для функции F, так как $\hat{F}(\Lambda) = F(\Lambda) = \Lambda$. Общих позиций у пустого слова Λ со словом $\hat{F}(s)$ нет, следовательно, не имеется и общих абсолютно различимых позиций.

2. Далее, $|\hat{F}(s)| = |F(\Lambda)F(s)| = |\Lambda F(s)| = |F(s)| > |\hat{F}(\Lambda)| = |F(\Lambda)| = 0$. Таким образом, пара (s, Λ) не является аномалией 2-го рода для функции F. Очевидно также, что данная пара не содержит префиксным образом аномалию 2-го рода для функции F, поскольку для пустого слова Λ не существует префиксов, отличных от самого пустого слова.

3. Слово $F(s)$ является суффиксом слова $\hat{F}(s) = F(\Lambda)F(s) = F(s)$. Запишем длину: $|F(s)| = |\hat{F}(s)| = |\hat{F}(s)| - 0 = |\hat{F}(s)| - |\hat{F}(\Lambda)|$.

Сопоставляя выводы п. 1–3, получаем, что перебираемая сущность $(q_s, q_\Lambda, F(s), s, \Lambda)$ регулярна. ■

Утверждение 7. Пусть перебираемая сущность $e_1 = (q_{\hat{s}_1}, q_{\hat{s}_2}, \alpha, \hat{s}_1, \hat{s}_2)$ регулярна и под действием пары символов (s_1, s_2) переходит в другую перебираемую сущность $e_2 = (q_{\hat{s}_1 s_1}, q_{\hat{s}_2 s_2}, \alpha', \hat{s}_1 s_1, \hat{s}_2 s_2)$. Тогда e_2 — регулярная сущность.

Доказательство. Выпишем некоторые условия регулярности сущности e_1. Во-первых, так как $(\hat{s}_1, \hat{s}_2)$ не является аномалией 1-го рода для функции F, слова $\hat{F}(\hat{s}_1)$ и $\hat{F}(\hat{s}_2)$ не имеют общей абсолютно различимой позиции. Во-вторых, α — суффикс $\hat{F}(\hat{s}_1)$ и $|\alpha| = |\hat{F}(\hat{s}_1)| - |\hat{F}(\hat{s}_2)| > 0$, поскольку $(\hat{s}_1, \hat{s}_2)$ не является аномалией 2-го рода для функции F. Следовательно, существует слово α_1, такое, что $|\alpha_1| = |\hat{F}(\hat{s}_2)|$ и $\hat{F}(\hat{s}_1) = \alpha_1\alpha$.

Пара $(\hat{s}_1, \hat{s}_2)$ является одноходовой. Следовательно, одноходовой является также пара $(\hat{s}_1 s_1, \hat{s}_2 s_2)$.

Проверим далее условия регулярности для сущности e_2.

1. $(\hat{s}_1 s_1, \hat{s}_2 s_2)$ не содержит префиксным образом аномалию для функции F:

— Под действием (s_1, s_2) сущность e_1 не переходит в аномалию 2-го рода для функции F, т. е. $|\alpha F(\hat{s}_1 s_1)| = |\alpha\psi(q_{\hat{s}_1}, s_1)| > |\psi(q_{\hat{s}_2}, s_2)| = |F(\hat{s}_2 s_2)|$. Следовательно, $|\hat{F}(\hat{s}_1)| - |\hat{F}(\hat{s}_2)| = |\alpha| > |F(\hat{s}_2 s_2)| - |F(\hat{s}_1 s_1)|$.

— Перенося слагаемые в неравенстве, получаем $|\hat{F}(\hat{s}_1 s_1)| = |\hat{F}(\hat{s}_1)| + |F(\hat{s}_1 s_1)| > |\hat{F}(\hat{s}_2)| + |F(\hat{s}_2 s_2)| = |\hat{F}(\hat{s}_2 s_2)|$. Таким образом, пара $(\hat{s}_1 s_1, \hat{s}_2 s_2)$ не является аномалией 2-го рода для функции F.

— Так как сущность e регулярна, пара $(\hat{s}_1, \hat{s}_2)$ не содержит префиксным образом аномалию 2-го рода для функции F. Кроме того, пара $(\hat{s}_1 s_1, \hat{s}_2 s_2)$ не является аномалией 2-го рода для функции F. В соответствии с утверждением 5 пара $(\hat{s}_1 s_1, \hat{s}_2 s_2)$ не содержит префиксным образом аномалию 2-го рода для функции F.

2. $(\hat{s}_1 s_1, \hat{s}_2 s_2)$ не является аномалией 1-го рода для функции F:

— Под действием пары символов (s_1, s_2) первая сущность не переходит в аномалию 1-го рода для функции F. Следовательно, слова $\alpha F(\hat{s}_1 s_1) = \alpha\psi(q_{\hat{s}_1}, s_1)$ и $F(\hat{s}_2 s_2) = \psi(q_{\hat{s}_2}, s_2)$ не имеют общей абсолютно различимой позиции.

— Предположим, что $(\hat{s}_1 s_1, \hat{s}_2 s_2)$ является аномалией 1-го рода для функции F, т. е. слова $\hat{F}(\hat{s}_1 s_1) = \alpha_1\alpha F(\hat{s}_1 s_1)$ и $\hat{F}(\hat{s}_2 s_2) = \hat{F}(\hat{s}_2)F(\hat{s}_2 s_2)$ имеют общую абсолютно различимую позицию.

— Так как $|\alpha_1| = |\hat{F}(\hat{s}_2)|$, а слова $\alpha F(\hat{s}_1)$ и $F(\hat{s}_2 s_2)$ не имеют общей абсолютно различимой позиции, то общая абсолютно различимая позиция является общей абсолютно различимой позицией слов $\alpha_1, \hat{F}(\hat{s}_2)$.

— Следовательно, она же является абсолютно различимой позицией слов $\hat{F}(\hat{s}_1) = \alpha_1\alpha$, $\hat{F}(\hat{s}_2)$, что заведомо исключено.

3. Слово α' — суффикс $\hat{F}(\hat{s}_1 s_1)$ и $|\alpha'| = |\hat{F}(\hat{s}_1 s_1)| - |\hat{F}(\hat{s}_2 s_2)|$:

— По условию, сущность e_1 под действием (s_1, s_2) переходит в e_2. Следовательно, α' — суффикс $\alpha F(\hat{s}_1 s_1) = \alpha\psi(q_{\hat{s}_1}, s_1)$, $|\alpha'| = |\alpha F(\hat{s}_1 s_1)| - |F(\hat{s}_2 s_2)|$.

— α — суффикс $\hat{F}(\hat{s}_1)$. Следовательно, $\alpha F(\hat{s}_1 s_1)$ — суффикс $F(\hat{s}_1)F(\hat{s}_1 s_1) = \hat{F}(\hat{s}_1 s_1)$.

— α' — суффикс $\alpha F(\hat{s}_1 s_1)$, $\alpha F(\hat{s}_1 s_1)$ — суффикс $\hat{F}(\hat{s}_1 s_1)$. Следовательно, α' — суффикс $\hat{F}(\hat{s}_1 s_1)$.

— Подсчитаем длину: $|\alpha'| = |\alpha F(\hat{s}_1 s_1)| - |F(\hat{s}_2 s_2)| = |\alpha| + |F(\hat{s}_1 s_1)| - |F(\hat{s}_2 s_2)| = |\hat{F}(\hat{s}_1)| - |\hat{F}(\hat{s}_2)| + |F(\hat{s}_1 s_1)| - |F(\hat{s}_2 s_2)| = |\hat{F}(\hat{s}_1 s_1)| - |\hat{F}(\hat{s}_2 s_2)|$.

Утверждение 7 доказано. ■

Утверждение 8. Пусть e_1 — регулярная перебираемая сущность. Тогда найдутся регулярная сущность e_2 и пара символов (s_1, s_2) алфавита S, такие, что e_2 переходит в e_1 под действием (s_1, s_2).

Доказательство.

1. Сущность e_1 имеет вид $(q_{\hat{s}_1 s_1}, q_{\hat{s}_2 s_2}, \alpha', \hat{s}_1 s_1, \hat{s}_2 s_2)$, где s_1, s_2 — символы алфавита S.

2. Выпишем некоторые условия регулярности сущности e_1. Во-первых, слово α' является суффиксом слова $\hat{F}(\hat{s}_1 s_1)$ и $|\alpha'| = |\hat{F}(\hat{s}_1 s_1)| - |\hat{F}(\hat{s}_2 s_2)|$. Следовательно, найдётся слово α'_1, такое, что $|\alpha'_1| = |\hat{F}(\hat{s}_2 s_2)|$ и $\hat{F}(\hat{s}_1 s_1) = \alpha'_1 \alpha'$.

3. Во-вторых, пара слов $(\hat{s}_1 s_1, \hat{s}_2 s_2)$ не является аномалией 1-го рода для функции F, т. е. слова $\hat{F}(\hat{s}_1 s_1)$ и $\hat{F}(\hat{s}_2 s_2)$ не имеют общей абсолютно различимой позиции.

4. Пара слов $(\hat{s}_1 s_1, \hat{s}_2 s_2)$ является одноходовой, т. е. $|\hat{s}_1 s_1| = |\hat{s}_2 s_2| + 1$. Следовательно, $|\hat{s}_1| = |\hat{s}_2| + 1$, т. е. $(\hat{s}_1, \hat{s}_2)$ также является одноходовой парой.

5. Так как $\hat{s}_1$ — префикс $\hat{s}_1 s_1$ и $\hat{s}_2$ — префикс $\hat{s}_2 s_2$, то в соответствии с утверждением 3 $(\hat{s}_1, \hat{s}_2)$ также не является аномалией 1-го рода для функции F.

6. Так как $(\hat{s}_1 s_1, \hat{s}_2 s_2)$ не является аномалией 2-го рода для функции F, то $|\hat{F}(\hat{s}_1 s_1)| > |\hat{F}(\hat{s}_2 s_2)|$. Следовательно, $|\alpha'| > 0$.

7. Так как пара $(\hat{s}_1 s_1, \hat{s}_2 s_2)$ не содержит префиксным образом аномалию 2-го рода для функции F, $\hat{s}_1$ — префикс $\hat{s}_1 s_1$, а $\hat{s}_2$ — префикс $\hat{s}_2 s_2$, то в соответствии с утверждением 4 пара $(\hat{s}_1, \hat{s}_2)$ также не содержит префиксным образом аномалию 2-го рода для функции F.

8. То, что $(\hat{s}_1, \hat{s}_2)$ — не аномалия 2-го рода для функции F, означает выполнение неравенства $|\hat{F}(\hat{s}_1)| > |\hat{F}(\hat{s}_2)|$. Следовательно, найдётся слово $\alpha \in A^*$, такое, что α — суффикс $\hat{F}(\hat{s}_1)$ и $|\alpha| = |\hat{F}(\hat{s}_1)| - |\hat{F}(\hat{s}_2)|$. Отсюда следует существование слова α_1, такого, что $|\alpha_1| = |\hat{F}(\hat{s}_2)|$ и $\hat{F}(\hat{s}_1) = \alpha_1 \alpha$.

9. Рассмотрим сущность $e_2 = (q_{\hat{s}_1}, q_{\hat{s}_2}, \alpha, \hat{s}_1, \hat{s}_2)$. Выводы п. 3, 4, 6 и 7 означают, что она регулярна. Докажем теперь, что под действием пары символов (s_1, s_2) она переходит в сущность e_1.

10. Перепишем тождества из п. 1 и 7: $\alpha'_1 \alpha' = \hat{F}(\hat{s}_1 s_1) = \hat{F}(\hat{s}_1) F(\hat{s}_1 s_1) = \alpha_1 \alpha F(\hat{s}_1 s_1)$. Таким образом, $|\alpha'| = |\alpha F(\hat{s}_1 s_1)| + |\alpha_1| - |\alpha'_1| = |\alpha F(\hat{s}_1 s_1)| + |\hat{F}(\hat{s}_2)| - |\hat{F}(\hat{s}_2 s_2)| = |\alpha F(\hat{s}_1 s_1)| - |F(\hat{s}_2 s_2)| < |\alpha F(\hat{s}_1 s_1)|$. Следовательно, α' — суффикс $\alpha F(\hat{s}_1 s_1)$.

11. Соотнося полученное в п. 9 выражение для $|\alpha'|$ с выводом п. 5, получаем $|\alpha\psi(q_{\hat{s}_1}, s_1)| = |\alpha F(\hat{s}_1 s_1)| > |F(\hat{s}_2 s_2)| = |\psi(q_{\hat{s}_2}, s_2)|$. Следовательно, сущность e_2 под действием (s_1, s_2) не переходит в аномалию 2-го рода.

12. Предположим, что e_2 под действием (s_1, s_2) переходит в аномалию 1-го рода, т. е. что слова $\alpha\psi(\hat{s}_1, s_1) = \alpha F(\hat{s}_1 s_1)$ и $\psi(q_{\hat{s}_2}, s_2) = F(\hat{s}_2 s_2)$ имеют общую абсолютно различимую позицию.

13. Так как $\hat{F}(\hat{s}_1 s_1) = \hat{F}(\hat{s}_1) F(\hat{s}_1 s_1) = \alpha_1 \alpha F(\hat{s}_1 s_1)$, $\hat{F}(\hat{s}_2 s_2) = \hat{F}(\hat{s}_2) F(\hat{s}_2 s_2)$ и $|\alpha_1| = |\hat{F}(\hat{s}_2)|$, то данная абсолютно различимая позиция является абсолютно различимой позицией слов $\hat{F}(\hat{s}_1 s_1)$ и $\hat{F}(\hat{s}_2 s_2)$, что противоречит выводу п. 2. Таким образом, e_2 не переходит в аномалию 1-го рода.

14. Соберём вместе выводы п. 9, 10 и 12: сущность e_2 под действием (s_1, s_2) не переходит ни в аномалию 1-го рода, ни в аномалию 2-го рода, α' является суффиксом $\alpha F(\hat{s}_1 s_1) = \alpha\psi(q_{\hat{s}_1}, s_1)$, $|\alpha'| = |\alpha F(\hat{s}_1 s_1)| - |F(\hat{s}_2 s_2)| = |\alpha\psi(q_{\hat{s}_1}, s_1)| - |\psi(q_{\hat{s}_2}, s_2)|$. Та-

ким образом, сущность e_2 под действием (s_1, s_2) переходит в сущность e_1, что и было заявлено в п. 8.

Утверждение 8 доказано. ■

3.2. Свойства обработки

Алгоритмом могут обрабатываться только регулярные сущности. Докажем, что каждая регулярная сущность может обрабатываться не более одного раза и что последовательность добавляемых в очередь сущностей упорядочена по размеру. Более того, по размеру упорядочена последовательность сущностей, подаваемых как аргумент в процедуру размещения в очереди.

Утверждение 9. Две подобные сущности не могут быть добавлены в очередь в разное время и, следовательно, не могут в разное время подаваться как аргумент в процедуру обработки.

Доказательство. Предположим обратное: пусть по ходу работы алгоритма две подобные сущности e_1 и e_2 добавлялись в очередь в разное время.

В очередь сущности попадают лишь в результате вызова процедуры размещения. Таким образом, сущности e_1 и e_2 подавались как аргумент на вход процедуры размещения в очереди и в результате её выполнений были добавлены в очередь. Для определённости предположим, что сущность e_1 размещалась ранее сущности e_2.

В соответствии с описанием процедуры размещения после завершения её работы на аргументе e_1 сущность e_1 находится в множестве принятых к обработке. Таким образом, она находится в множестве принятых к обработке во время работы процедуры размещения на аргументе e_2. Так как сущности e_1 и e_2 подобны, то e_2 не будет добавлена в очередь — противоречие. ■

Утверждение 10. В очередь алгоритма и в множество принятых к обработке могут попасть только регулярные сущности. Следовательно, все сущности, поступающие на вход процедуры обработки, регулярны.

Доказательство. После выполнения процедуры инициализации в очереди находятся инициальные сущности, которые регулярны согласно утверждению 6.

Всякая сущность, попадающая в очередь, является результатом действия пары (s_1, s_2) на сущность, которая была передана как аргумент в функцию обработки и, следовательно, сама попадала в очередь. Это действие, согласно утверждению 7, переводит регулярные сущности в регулярные. По индукции заключаем, что все попавшие в очередь сущности регулярны.

В множество принятых к обработке попадают те же сущности, что и в очередь. Следовательно, все они регулярны. ■

Утверждение 11. В каждый момент времени в очереди находятся или сущности одного и того же размера k, или сущности двух последовательных размеров k и $k+1$, причём очередь упорядочена по размеру, т. е. сущности размера $k+1$ находятся в очереди после сущностей размера k.

Доказательство. Во время выполнения процедуры инициализации и после её завершения находящиеся в очереди сущности имеют размер 0.

Пусть указанное в условии свойство выполнено перед обработкой очередной полной сущности с головы очереди. Докажем, что оно выполнено и после обработки данной сущности. По индукции заключаем, что это свойство выполняется всегда.

Действительно, пусть в очереди все сущности имеют размер k. Тогда, обрабатывая сущность с головы, процедура либо ничего не добавляет в очередь, либо добавляет в хвост очереди сущности размера $k+1$, что сохраняет выполнение свойства.

Если в очереди находятся сущности двух размеров k и $k+1$, то первой в очереди находится сущность размера k. Обрабатывая её, процедура, может быть, добавляет в хвост очереди сущности размера $k+1$, что также сохраняет выполнение свойства. ■

Утверждение 12. Пусть сущность e_1 размера k_1 и сущность e_2 размера k_2 попадают в очередь. Тогда если $k_1 < k_2$, то e_1 попадает в очередь ранее сущности e_2. И, следовательно, если обе сущности e_1 и e_2 подаются как аргумент в процедуру обработки, то сущность e_1 подаётся ранее сущности e_2.

Доказательство. Рассмотрим последовательность всех сущностей, которые попадают в очередь. Предположим, что в ней нарушен порядок по размеру. Не уменьшая общности, можно предположить, что порядок нарушен для двух соседних сущностей, т. е. имеются две сущности e_1 и e_2, такие, что $\mathrm{size}(e_1) > \mathrm{size}(e_2)$ и сущность e_2 была добавлена в очередь сразу вслед за сущностью e_1. Отметим, что сущности e_1 и e_2 не могут одновременно присутствовать в очереди, поскольку по утверждению 11 очередь упорядочена по размеру в каждый момент времени.

Предположим, что сущность e_2 добавляется в очередь во время выполнения процедуры инициализации. Тогда e_2 является инициальной сущностью и $\mathrm{size}(e_2) = 0$. Так как сущность e_1 добавлена в очередь ранее сущности e_2, то сущность e_1 также добавлена во время выполнения процедуры инициализации. Следовательно, $\mathrm{size}(e_1) = 0$, что противоречит условию $\mathrm{size}(e_1) > \mathrm{size}(e_2)$. Таким образом, e_2 добавляется в очередь не во время выполнения процедуры инициализации, а во время обработки некой сущности e. В момент завершения обработки сущности e сущность e_2 присутствует в очереди, и, следовательно, в данный момент сущность e_1 в очереди уже не присутствует.

Так как сущность e_2 была добавлена в очередь сразу вслед за сущностью e_1, то в данный момент сущность e_2 находится в голове очереди и в последовательности сущностей, которые попадают в очередь, сущность e непосредственно предшествует сущности e_2. Таким образом, $e = e_1$.

Так как e_2 добавляется в очередь во время обработки сущности e_1, то $\mathrm{size}(e_2) = \mathrm{size}(e_1) + 1$, что противоречит условию $\mathrm{size}(e_1) > \mathrm{size}(e_2)$. ■

Утверждение 13. Пусть сущность e_1 размера k_1 и сущность e_2 размера k_2 подаются как аргумент в процедуру размещения в очереди. Тогда если $k_1 < k_2$, то вызов процедуры размещения в очереди с аргументом e_1 предшествует вызову данной процедуры с аргументом e_2.

Доказательство. Поскольку $k_1 < k_2$, то $e_1 \neq e_2$. Следовательно, вызовы процедуры размещения в очереди с аргументами e_1 и e_2 не могут происходить одновременно.

Предположим обратное. Пусть вызов процедуры размещения в очереди с аргументом e_1 последовал за вызовом с аргументом e_2.

Предположим, что процедура размещения в очереди с аргументом e_2 была вызвана процедурой инициализации. Тогда $k_1 < 0 = \mathrm{size}(e_2) = k_2$, что невозможно, поскольку размер k_1 сущности e_1 не может быть отрицательным. Таким образом, процедура размещения в очереди с аргументом e_2 не была вызвана процедурой инициализации. В соответствии с предположением процедура инициализации не вызывала процедуру размещения в очереди также и с аргументом e_1.

Следовательно, найдутся сущности e'_1 и e'_2, такие, что вызовы процедуры размещения в очереди с аргументами e_1 и e_2 произошли во время выполнения процедуры обработки с аргументами e'_1 и e'_2 соответственно.

Таким образом, $\text{size}(e'_1) = \text{size}(e_1) - 1 = k_1 - 1$, $\text{size}(e'_2) = \text{size}(e_2) - 1 = k_2 - 1$. Отсюда следует $\text{size}(e'_1) < \text{size}(e'_2)$. В соответствии с утверждением 12 заключаем, что сущность e'_1 подается как аргумент в процедуру обработки ранее сущности e'_2.

Следовательно, вызов процедуры размещения в очереди с аргументом e_1, происходящий во время обработки e'_1, произошёл ранее вызова процедуры размещения в очереди с аргументом e_2, происходящего во время обработки e'_2, что противоречит предположению. ■

Утверждение 14. Пусть перебираемая сущность e размера k подаётся как аргумент в процедуру размещения в очереди. Тогда после выполнения данной процедуры в множестве принятых к обработке находится сущность e_1 размера не более k, подобная сущности e.

Доказательство. Предположим обратное: пусть после выполнения данной процедуры в множестве принятых к обработке нет сущности, подобной сущности e и имеющий размер не более k.

Предположим, что до выполнения процедуры в множестве принятых к обработке нет сущности, подобной сущности e. Тогда данная процедура добавляет в очередь и в множество принятых к обработке саму сущность e, и после её выполнения e находится в множестве принятых к обработке. Сущность e подобна самой себе и имеет размер не более k, что противоречит предположению. Таким образом, до выполнения процедуры размещения в очереди в множестве принятых к обработке лежит некая сущность e_1, подобная сущности e.

Сущность e_1 лежит в множестве принятых к обработке и после выполнения данной процедуры. Следовательно, $\text{size}(e_1) > \text{size}(e)$.

Так как в множество принятых к обработке попадают только сущности, добавляемые в очередь, а сущности, добавляемые в очередь, подавались как аргумент в процедуру размещения в очереди, то сущность e_1 подавалась как аргумент в процедуру размещения в очереди. Таким образом, вызов процедуры размещения в очереди с аргументом e_1 предшествует вызову с аргументом e. Применяя утверждение 13, получаем $\text{size}(e_1) \leqslant \text{size}(e)$ — противоречие. ■

3.3. Исключения

Установим свойства алгоритма, связанные с теми случаями, когда его работа оканчивается выдачей исключения.

Определение 19. Будем говорить, что алгоритм k-прерывен, если он выдаёт исключение при выполнении обработки некой перебираемой сущности размера k', где $k' \leqslant k$.

Замечание 1. Выражения «сущность подаётся как аргумент в процедуру обработки» и «сущность обрабатывается алгоритмом» различаются по смыслу. Если сущность обрабатывается, то она перед этим подаётся как аргумент. Однако обратное неверно, так как во время обработки поданной как аргумент сущности алгоритм может выдать исключение. В данном случае сущность не считается обработанной.

Утверждение 15. Пусть алгоритм k-прерывен. Тогда в процедуру обработки в качестве аргумента не может подаваться никакая сущность размера больше k.

Доказательство. Предположим обратное. Пусть e_1 — сущность размера не больше k, при обработке которой выдаётся исключение, а e_2 — сущность размера больше k, которая поступает как аргумент в процедуру обработки. Поскольку $\mathrm{size}(e_1) < \mathrm{size}(e_2)$, по утверждению 12 получаем, что сущность e_2 подавалась как аргумент в процедуру переработки позже, чем сущность e_1. Однако при обработке сущности e_1 алгоритм выдаёт исключение и завершает работу. Противоречие. ■

Утверждение 16. Пусть алгоритм не является k-прерывным и перебираемая сущность e размера не больше k попадает в очередь. Тогда сущность e обрабатывается алгоритмом.

Доказательство. Предположим обратное: сущность e не обрабатывается алгоритмом. Поскольку она попадает в очередь, то алгоритм выдаёт исключение.

Пусть алгоритм выдаёт исключение при обработке перебираемой сущности e_1. Поскольку сущность e не обрабатывается, то она остаётся не обработанной и в момент, непосредственно предшествующий подаче сущности e_1 как аргумента в процедуру обработки. Возможен в том числе и случай $e = e_1$.

В описанный момент сущность e_1 находится в голове очереди. Поскольку сущность e попадает в очередь, но не обрабатывается, то в данный момент она находится в очереди. Следовательно, $\mathrm{size}(e_1) \leqslant \mathrm{size}(e)$, поскольку очередь в каждый момент упорядочена по размеру сущностей (см. утверждение 11).

Таким образом, $\mathrm{size}(e_1) \leqslant \mathrm{size}(e) \leqslant k$ и алгоритм выдаёт исключение при обработке сущности e_1 размера не больше k. То есть алгоритм k-прерывен, что противоречит условию. ■

Утверждение 17. Пусть алгоритм не k-прерывен, e_1 — регулярная сущность размера k. Тогда найдётся регулярная сущность e_2 размера не больше k, подобная сущности e_1 и обрабатывающаяся алгоритмом.

Доказательство. Докажем по индукции.

Б а з а и н д у к ц и и : $k = 0$.

Пусть алгоритм не 0-прерывен, e_1 — регулярная сущность размера 0, т. е. инициальная.

Процедура инициализации подаёт на размещение в очереди все инициальные сущности, в том числе e_1. В соответствии с утверждением 14 после вызова процедуры размещения в очереди в множестве принятых к обработке лежит некая сущность e_2, подобная сущности e_1 и имеющая размер не более 0. Сущность не может иметь отрицательный размер, т. е. $\mathrm{size}(e_2) = 0$.

Так как e_2 попадает в множество принятых к обработке, то e_2 попадает и в очередь. В соответствии с утверждением 16 сущность e_2 обрабатывается алгоритмом, так как алгоритм не 0-прерывен и $\mathrm{size}(e_2) = 0$. Сущность e_2 является регулярной (см. утверждение 10).

Ш а г и н д у к ц и и. Пусть утверждение верно для некоторого k. Тогда оно верно и для $k + 1$.

Пусть алгоритм не $(k + 1)$-прерывен, e_1 — регулярная сущность размера $k + 1$.

По утверждению 8 найдутся регулярная сущность e_1' размера k и пара символов (s_1, s_2) алфавита S, такие, что e_1' под действием (s_1, s_2) переходит в e_1.

Так как алгоритм не $(k + 1)$-прерывен, то он и не k-прерывен. Применяя предположение индукции к регулярной сущности e_1' размера k, получаем, что найдётся регулярная сущность e_2' размера не больше k, подобная сущности e_1' и обрабатывающаяся алгоритмом.

В соответствии с утверждением 2, так как сущности e'_2 и e'_1 подобны и под действием (s_1, s_2) сущность e'_1 переходит в e_1, то найдётся сущность e_2, такая, что e'_2 под действием (s_1, s_2) переходит в e_2. Кроме того, $\mathrm{size}(e_2) \leqslant k+1$.

В соответствии с утверждением 1, так как сущность e'_1 подобна сущности e'_2 и под действием (s_1, s_2) сущности e'_1 и e'_2 переходят в e_1 и e_2 соответственно, то сущности e_1 и e_2 также являются подобными.

Так как e'_2 обрабатывается алгоритмом и под действием (s_1, s_2) сущность e'_2 переходит в e_2, то во время обработки e'_2 сущность e_2 подается как аргумент в процедуру размещения в очереди. В соответствии с утверждением 14 после вызова этой процедуры в множестве принятых к обработке лежит некая сущность e_3, подобная сущности e_2 и имеющая размер не больше размера сущности e_2. Сущность e_3 является регулярной (см. утверждение 10).

Сущность e_3 подобна e_2, сущность e_2 подобна e_1. Следовательно, сущность e_3 подобна сущности e_1.

Так как $\mathrm{size}(e_3) \leqslant \mathrm{size}(e_2) \leqslant k+1$ и алгоритм не $(k+1)$-прерывен, то в соответствии с утверждением 16 сущность e_3 обрабатывается алгоритмом. ■

3.4. Проверка правильности

Докажем, что алгоритм выдаёт исключение тогда и только тогда, когда F не является правильной функцией. В соответствии с теоремой 1 следует установить, что алгоритм выдаёт исключение тогда и только тогда, когда существует одноходовая аномалия 2-го рода для функции F, не являющаяся аномалией 1-го рода для функции F.

Утверждение 18. Пусть алгоритм выдаёт исключение. Тогда найдётся одноходовая пара слов алфавита S, являющаяся аномалией 2-го рода, но не являющаяся аномалией 1-го рода для функции F.

Доказательство. Пусть исключение выдаётся во время обработки регулярной (см. утверждение 10) сущности $e = (q_{\hat{s}_1}, q_{\hat{s}_2}, \alpha, \hat{s}_1, \hat{s}_2)$. Выпишем некоторые условия её регулярности: $(\hat{s}_1, \hat{s}_2)$ — одноходовая пара, не являющаяся аномалией 1-го или 2-го рода для функции F, слово α — суффикс слова $\hat{F}(\hat{s}_1)$, и $|\alpha| = |\hat{F}(\hat{s}_1)| - |\hat{F}(\hat{s}_2)|$. Тогда найдётся слово α_1, такое, что $\hat{F}(\hat{s}_1) = \alpha_1\alpha$ и $|\alpha_1| = |\hat{F}(\hat{s}_2)|$.

Так как при обработке сущности e алгоритм выдаёт исключение, то существует пара символов (s_1, s_2), под действием которой сущность e переходит в аномалию 2-го рода, но не переходит в аномалию 1-го рода. Значит, выполнено неравенство $|\alpha F(\hat{s}_1 s_1)| = |\alpha\psi(q_{\hat{s}_1}, s_1)| \leqslant |\psi(q_{\hat{s}_2}, s_2)| = |F(\hat{s}_2 s_2)|$ и, следовательно, $|\hat{F}(\hat{s}_1 s_1)| = |\hat{F}(\hat{s}_1)| + |F(\hat{s}_1 s_1)| = |\alpha_1| + |\alpha| + |F(\hat{s}_1 s_1)| = |\alpha_1| + |\alpha F(\hat{s}_1 s_1)| \leqslant |\alpha_1| + |F(\hat{s}_2 s_2)| = |\hat{F}(\hat{s}_2)| + |F(\hat{s}_2 s_2)| = |\hat{F}(\hat{s}_2 s_2)|$. Таким образом, пара $(\hat{s}_1 s_1, \hat{s}_2 s_2)$ является аномалией 2-го рода для функции F.

Так как под действием (s_1, s_2) сущность e не переходит в аномалию 1-го рода, слова $\alpha F(\hat{s}_1 s_1)$ и $F(\hat{s}_2 s_2)$ не имеют общей абсолютно различимой позиции; так как $(\hat{s}_1, \hat{s}_2)$ не является аномалией 1-го рода для функции F, то и слова $\hat{F}(\hat{s}_1)$ и $\hat{F}(\hat{s}_2)$ не имеют общей абсолютно различимой позиции. Поскольку α_1 — префикс $\hat{F}(\hat{s}_1)$, то слова α_1 и $\hat{F}(\hat{s}_2)$ также не имеют общей абсолютно различимой позиции.

Так как $|\alpha_1| = |\hat{F}(\hat{s}_2)|$, а пары слов $(\alpha_1, \hat{F}(\hat{s}_2))$ и $(\alpha F(\hat{s}_1 s_1), F(\hat{s}_2 s_2))$ не имеют общей абсолютно различимой позиции, то абсолютно различимой позиции нет также у слов $\hat{F}(\hat{s}_1 s_1) = \hat{F}(\hat{s}_1)F(\hat{s}_1 s_1) = \alpha_1\alpha F(\hat{s}_1 s_1)$ и $\hat{F}(\hat{s}_2 s_2) = \hat{F}(\hat{s}_2)F(\hat{s}_2 s_2)$. Таким образом, пара слов $(\hat{s}_1 s_1, \hat{s}_2 s_2)$ не является аномалией 1-го рода для функции F.

Так как $(\hat{s}_1, \hat{s}_2)$ — одноходовая пара слов, то пара слов $(\hat{s}_1 s_1, \hat{s}_2 s_2)$ также является одноходовой. Таким образом, $(\hat{s}_1 s_1, \hat{s}_2 s_2)$ — одноходовая пара слов, являющаяся аномалией 2-го рода, но не являющаяся аномалией 1-го рода для функции F. ■

Утверждение 19. Пусть одноходовая пара слов $(\hat{s}_1, \hat{s}_2)$ является аномалией 2-го рода для функции F, но не является аномалией 1-го рода для функции F, и $|\hat{s}_2| = k$. Тогда алгоритм выдаёт исключение и является $(k-1)$-прерывным.

Доказательство. Предположим обратное, т. е. что алгоритм не является $(k-1)$-прерывным.

Пусть $(\hat{s}'_1, \hat{s}'_2)$ — минимальная по длине $|\hat{s}'_2|$ одноходовая пара слов, являющаяся аномалией 2-го рода, но не являющаяся аномалией 1-го рода для функции F. Полагаем далее $|\hat{s}_2| = k'$. В соответствии с минимальностью длины выполнено $k' \leqslant k$.

Поскольку для любого символа $s \in S$ верно $|\hat{F}(s)| = |F(\Lambda)F(s)| = |\Lambda F(s)| = |F(s)| > |F(\Lambda)| = 0$, то (s, Λ) не может быть аномалией 2-го рода для функции F. Следовательно, $|\hat{s}'_2| > 0$, $|\hat{s}'_1| = |\hat{s}'_2| + 1 > 0$. Найдутся слова $\hat{s}'_1$ и $\hat{s}'_2$ и символы s_1 и s_2 алфавита S, такие, что $\hat{s}'_1 = \hat{s}''_1 s_1$, $\hat{s}'_2 = \hat{s}''_2 s_2$. Также выполнено $|\hat{s}''_2| = |\hat{s}'_2| - 1 = k' - 1 \leqslant \leqslant k - 1$. Пара $(\hat{s}''_1, \hat{s}''_2)$, очевидно, является одноходовой.

Так как $\hat{s}''_1$ — префикс $\hat{s}'_1$, $\hat{s}''_2$ — префикс $\hat{s}'_2$ и пара слов $(\hat{s}'_1, \hat{s}'_2)$ не является аномалией 1-го рода для функции F, то в соответствии с утверждением 3 пара слов $(\hat{s}''_1, \hat{s}''_2)$ не является аномалией 1-го рода для функции F.

Предположим, что пара $(\hat{s}''_1, \hat{s}''_2)$ содержит префиксным образом аномалию 2-го рода для функции F. Тогда существует одноходовая пара $(\hat{s}'''_1, \hat{s}'''_2)$, являющаяся аномалией 2-го рода для функции F, такая, что $\hat{s}'''_1$ — префикс $\hat{s}''_1$, $\hat{s}'''_2$ — префикс $\hat{s}''_2$. Тогда по утверждению 3 пара $(\hat{s}'''_1, \hat{s}'''_2)$ не является аномалией 1-го рода для функции F.

Таким образом, одноходовая пара $(\hat{s}'''_1, \hat{s}'''_2)$ является аномалией 2-го рода, но не является аномалией 1-го рода для функции F, и $|\hat{s}'''_2| \leqslant |\hat{s}''_2| = |\hat{s}_1| - 1$, что противоречит минимальности выбора пары $(\hat{s}'_1, \hat{s}'_2)$. Это означает, что предположение ложно, т. е. пара $(\hat{s}''_1, \hat{s}''_2)$ не содержит префиксным образом аномалию 2-го рода для функции F и, следовательно, не является аномалией 2-го рода для функции F.

Так как одноходовая пара $(\hat{s}''_1, \hat{s}''_2)$ не является аномалией 2-го рода для функции F, то $|\hat{F}(\hat{s}''_1)| > |\hat{F}(\hat{s}''_2)|$. Следовательно, найдутся слова α, α_1, такие, что $\hat{F}(\hat{s}''_1) = \alpha_1 \alpha$, $|\alpha_1| = |\hat{F}(\hat{s}''_2)|$, $|\alpha| = |\hat{F}(\hat{s}''_1)| - |\hat{F}(\hat{s}''_2)|$.

Рассмотрим сущность $e = (q_{\hat{s}''_1}, q_{\hat{s}''_2}, \alpha, \hat{s}''_1, \hat{s}''_2)$. Она регулярна и имеет размер не больше $k - 1$. Так как алгоритм не $(k-1)$-прерывен, то в соответствии с утверждением 17 найдётся регулярная сущность e_1 размера не более $k - 1$, подобная сущности e и обрабатывающаяся алгоритмом.

Так как $(\hat{s}'_1, \hat{s}'_2)$ является аномалией 2-го рода для функции F, то выполнено неравенство $|\hat{F}(\hat{s}'_1)| - |\hat{F}(\hat{s}'_2)| \leqslant 0$. Произведём расчет длин: $|\hat{F}(\hat{s}'_1)| - |\hat{F}(\hat{s}'_2)| = (|\alpha_1| + \\ + |\alpha F(\hat{s}'_1)|) - (|\hat{F}(\hat{s}''_2)| + |F(\hat{s}'_2)|) = |\alpha F(\hat{s}'_1)| - |F(\hat{s}'_2)|$. Таким образом, $|\alpha\psi(q_{\hat{s}'_1}, s_1)| = \\ = |\alpha F(\hat{s}''_1 s_1)| = |\alpha F(\hat{s}'_1)| \leqslant |F(\hat{s}'_2)| = |F(\hat{s}''_2 s_2)| = |\psi(q_{\hat{s}''_2}, s_2)|$. Следовательно, под действием (s_1, s_2) сущность e переходит в аномалию 2-го рода.

Предположим, что под действием (s_1, s_2) сущность e переходит в аномалию 1-го рода. Тогда слова $\alpha\psi(q_{\hat{s}''_1}, s_1) = \alpha F(\hat{s}''_1 s_1) = \alpha F(\hat{s}'_1)$, $\psi(q_{\hat{s}''_2}, s_2) = F(\hat{s}''_2 s_2) = F(\hat{s}'_2)$ имеют общую абсолютно различимую позицию.

Запишем тождества: $\hat{F}(\hat{s}'_1) = \hat{F}(\hat{s}''_1 s_1) = \hat{F}(\hat{s}''_1) F(\hat{s}''_1 s_1) = \alpha_1 \alpha F(\hat{s}'_1)$, $\hat{F}(\hat{s}'_2) = \\ = \hat{F}(\hat{s}''_2 s_2) = \hat{F}(\hat{s}''_2) F(\hat{s}''_2 s_2) = \hat{F}(\hat{s}''_2) F(\hat{s}'_2)$.

Из равенства $|\alpha_1| = |\hat{F}(\hat{s}''_2)|$ следует, что абсолютно различимая позиция слов $\alpha F(\hat{s}'_1)$ и $F(\hat{s}'_2)$ является также абсолютно различимой позицией слов $\hat{F}(\hat{s}'_1)$ и $\hat{F}(\hat{s}'_2)$. Таким

образом, пара слов $(\hat{s}'_1, \hat{s}'_2)$ является аномалией 1-го рода для функции F, что противоречит её выбору. Это означает, что под действием (s_1, s_2) сущность e не переходит в аномалию 1-го рода.

Итак, под действием (s_1, s_2) сущность e переходит в аномалию 2-го рода, но не переходит в аномалию 1-го рода, и сущность e_1 подобна сущности e. В соответствии с утверждением 1 под действием (s_1, s_2) сущность e_1 также переходит в аномалию 2-го рода, но не переходит в аномалию 1-го рода. Значит, алгоритм выдаёт исключение во время обработки сущности e_1. Поскольку сущность e_1 имеет размер не более $k-1$, алгоритм $(k-1)$-прерывен — противоречие. ■

3.5. Недостижимые сущности

Осталось установить, что алгоритм не может работать бесконечно долго. Докажем, что множество обрабатываемых сущностей является конечным, т. е. процедура обработки может вызываться только конечное число раз.

Определение 20. Регулярная перебираемая сущность вида $(q_1, q_2, \alpha, \hat{s}_1 s'_1 s''_1, \hat{s}_2)$ называется недостижимой, если выполнено неравенство $|\hat{F}(\hat{s}_1)| \geqslant |\hat{F}(\hat{s}_2)|$.

Утверждение 20. Пусть $e = (q_1, q_2, \alpha, \hat{s}_1 s'_1 s''_1, \hat{s}_2)$ — регулярная недостижимая сущность. Тогда сущность e не добавляется в очередь.

Доказательство. Предположим обратное: сущность e добавляется в очередь. Обозначим: $k = \mathrm{size}(e) = |\hat{s}_2|$.

В силу регулярности пара $(\hat{s}_1 s'_1 s''_1, \hat{s}_2)$ является одноходовой, т. е. $|\hat{s}_1 s'_1 s''_1| = |\hat{s}_2| + 1$. Следовательно, $|\hat{s}_2| = |\hat{s}_1| + 1$. Таким образом, пара $(\hat{s}_2, \hat{s}_1)$ также является одноходовой.

Так как $k = |\hat{s}_2| = |\hat{s}_1| + 1$, то $k > 0$ и сущность e не является инициальной. Следовательно, она добавляется в очередь во время обработки некой регулярной сущности e_1, $\mathrm{size}(e_1) = \mathrm{size}(e) - 1 = k - 1$. Таким образом, сущность e_1 размера $k-1$ подаётся как аргумент в процедуру обработки.

В силу регулярности пара $(\hat{s}_1 s'_1 s''_1, \hat{s}_2)$ не является аномалией 1-го рода для функции F, т. е. слова $\hat{F}(\hat{s}_1 s'_1 s''_1), \hat{F}(\hat{s}_2)$ не имеют общей абсолютно различимой позиции. Так как $\hat{F}(\hat{s}_1)$ — префикс $\hat{F}(\hat{s}_1 s'_1 s''_1)$, то общей абсолютно различимой позиции не имеют также слова $\hat{F}(\hat{s}_1)$ и $\hat{F}(\hat{s}_2)$. Таким образом, пара $(\hat{s}_2, \hat{s}_1)$ также не является аномалией 1-го рода для функции F.

Так как $|\hat{F}(\hat{s}_1)| \geqslant |\hat{F}(\hat{s}_2)|$, пара $(\hat{s}_2, \hat{s}_1)$ является аномалией 2-го рода для функции F. Ввиду $|\hat{s}_1| = |\hat{s}_2| - 1 = k - 1$ и утверждения 19 это значит, что алгоритм является $(k-2)$-прерывным. Тогда в соответствии с утверждением 15 сущность e_1 не может подаваться как аргумент в процедуру обработки — противоречие. ■

Путём длины 2 автоматного графа называется последовательность вида $q_1 s_1 q_2 s_2 q_3$, состоящая из состояний $q_1, q_2, q_3 \in Q$ и символов $s_1, s_2 \in S$, таких, что $q_2 = \varphi(q_1, s_1)$, $q_3 = \varphi(q_2, s_2)$. Данному пути длины 2 ставится в соответствие слово $\psi(q_1, s_1)\psi(q_2, s_2)$. Определим множество P_2, состоящее из слов алфавита A, следующим образом: слово α лежит в множестве P_2 тогда и только тогда, когда существует путь длины 2, поставленный в соответствие данному слову α. Путь длины 2 однозначно задаётся первым состоянием q_1 и символами s_1, s_2. Следовательно, выполняется неравенство $|P_2| \leqslant |Q||S|^2$. Каждое слово из P_2 получено конкатенацией двух слов, являющихся значениями выходной функции автомата. Таким образом, длина слов из P_2 не превышает $2L$.

Будем говорить, что слово β_1 является строгим суффиксом слова β_2, если β_1 — суффикс β_2 и $\beta_1 \neq \beta_2$. Множество слов, являющихся непустыми строгими суффик-

сами слов из множества P_2, обозначим P_*. Количество непустых строгих суффиксов каждого слова из P_2 не превышает $2L-1$. Следовательно, справедлива оценка $|P_*| \leqslant (2L-1)|Q||S|^2$.

Утверждение 21. Пусть $e = (q_{\hat{s}_1 s'_1 s_1}, q_{\hat{s}_2 s_2}, \alpha, \hat{s}_1 s'_1 s_1, \hat{s}_2 s_2)$ — регулярная сущность, не являющаяся недостижимой и не являющаяся инициальной. Тогда $\alpha \in P_*$.

Доказательство. Выпишем некоторые условия регулярности: $(\hat{s}_1 s'_1 s_1, \hat{s}_2 s_2)$ не является аномалией 2-го рода для функции F, т. е. $|\hat{F}(\hat{s}_2 s_2)| < |\hat{F}(\hat{s}_1 s'_1 s_1)|$. Слово α — суффикс слова $\hat{F}(\hat{s}_1 s'_1 s_1)$ и $|\alpha| = |\hat{F}(\hat{s}_1 s'_1 s_1)| - |\hat{F}(\hat{s}_2 s_2)| > 0$. Следовательно, существует слово α_1, такое, что $\hat{F}(\hat{s}_1 s'_1 s_1) = \alpha_1 \alpha$, $|\alpha_1| = |\hat{F}(\hat{s}_2 s_2)|$.

Так как сущность $(q_{\hat{s}_1 s'_1 s_1}, q_{\hat{s}_2 s_2}, \alpha, \hat{s}_1 s'_1 s_1, \hat{s}_2 s_2)$ не является недостижимой, то $|\hat{F}(\hat{s}_1)| < |\hat{F}(\hat{s}_2 s_2)| = |\alpha_1|$. Положим $\alpha' = F(\hat{s}_1 s'_1) F(\hat{s}_1 s'_1 s_1) = \psi(q_{\hat{s}_1}, s'_1)\psi(q_{\hat{s}_1 s'_1}, s_1)$. Слово α' соответствует пути длины 2 $q_{\hat{s}_1} s'_1 q_{\hat{s}_1 s'_1} s_1 q_{\hat{s}_1 s'_1 s_1}$ автоматного графа. Таким образом, $\alpha' \in P^2$.

Выпишем тождество: $\alpha_1 \alpha = \hat{F}(\hat{s}_1 s'_1 s_1) = \hat{F}(\hat{s}_1) F(\hat{s}_1 s'_1) F(\hat{s}_1 s'_1 s_1) = \hat{F}(\hat{s}_1)\alpha'$. Так как $|\hat{F}(\hat{s}_1)| < |\alpha_1|$, слово α является строгим непустым суффиксом слова $\alpha' \in P_2$. Следовательно, $\alpha \in P_*$. ■

3.6. Оценка сложности

Используя полученные результаты, оценим сверху количество запусков процедуры обработки (и тем самым покажем, что оно конечно), а также получим оценку сложности алгоритма.

Утверждение 22. В очередь попадает не более $|S| + (2L-1)|Q|^3|S|^2$ сущностей. Следовательно, процедура обработки запускается не более $|S| + (2L-1)|Q|^3|S|^2$ раз.

Доказательство.

1. Процедура инициализации добавляет в очередь не более $|S|$ инициальных сущностей.

2. Рассмотрим множество регулярных сущностей, не являющихся недостижимыми и не являющихся инициальными. Каждая такая сущность имеет вид $(q_1, q_2, \alpha, \hat{s}_1, \hat{s}_2)$, где $q_1, q_2 \in Q$; $\alpha \in A^*$; $\hat{s}_1, \hat{s}_2 \in S^*$. В соответствии с утверждением 21 выполнено $\alpha \in P_*$.

3. Разделим данное множество на классы следующим образом: две сущности принадлежат одному классу, если они подобны, т. е. совпадают их первые три компоненты. Первые две компоненты могут принимать не более $|Q|$ различных значений, а третья компонента — не более $|P_*|$ значений. Таким образом, число классов разбиения конечно и не превышает $|Q|^2|P_*| \leqslant |Q|^2(2L-1)|Q||S|^2 = (2L-1)|Q|^3|S|^2$.

4. В соответствии с утверждениями 10 и 20 в очередь могут добавляться только инициальные сущности и сущности из множества п. 2. Согласно утверждению 9, в очередь может быть добавлена не более чем одна сущность из каждого класса разбиения. Следовательно, всего в очередь попадает не более $|S| + (2L-1)|Q|^3|S|^2$ сущностей.

Утверждение 22 доказано. ■

Утверждение 23. Возможность бесконечно долгой работы алгоритма исключена. Если F — правильная функция, то алгоритм остановится в состоянии пустой очереди, иначе он выдаст исключение. Сложность алгоритма составляет $O(L|Q|^3|S|^4)$.

Доказательство. В соответствии с теоремой 1 правильность функции F эквивалентна отсутствию аномалии 2-го рода для функции F, не являющейся аномалией 1-го рода для функции F.

В соответствии с утверждениями 18 и 19 исключение выдаётся тогда и только тогда, когда таковая аномалия присутствует, т. е. F не является правильной функцией.

В соответствии с утверждением 22 процедура обработки может быть запущена не более $|S| + (2L-1)|Q|^3|S|^2 \leqslant |S| + 2L|Q|^3|S|^2$ раз. Таким образом, возможность бесконечно долгой работы алгоритма исключена и, если функция F правильна, то алгоритм завершает свою работу в состоянии пустой очереди.

Проверка перехода в аномалии 1-го и 2-го рода, расчёт перехода в другую сущность, а также проверка наличия подобной сущности в множестве принятых к обработке занимает константное время. Следовательно, сложность процедуры обработки составляет $O(|S|^2)$.

Процедура предынициализации имеет сложность $O(1) + O(|Q||S|)$, процедура инициализации — $O(|S|)$.

Таким образом, сложность алгоритма составляет $O(1) + O(|Q||S|) + O(|S|) + O(|S|^2)(|S| + 2L|Q|^3|S|^2) = O(|S|) + O(|S|^3) + O(L|Q|^3|S|^4) = O(L|Q|^3|S|^4)$. ■

Формулировка утверждения 23 тождественна формулировке теоремы 4. Теорема 4 доказана.

Заключение

Представлен алгоритм, позволяющий проверить принадлежность функции F классу правильных за полиномиальное время от числа состояний $|Q|$ автомата, представляющего соответствующую ограниченно-детерминированную функцию $\hat{F}$, числа символов $|S|$ входного алфавита и максимальной длины L выходного слова.

Алгоритм хранит в очереди и в множестве принятых к обработке перебираемые сущности, представляющие собой пятёрки $(q_1, q_2, \alpha, \hat{s}_1, \hat{s}_2)$. Однако две последние компоненты были введены в состав сущности только с целью провести доказательство корректности работы алгоритма, а на практике их хранение в памяти является излишним.

Утверждения 1 и 2 гарантируют, что хранить в очереди и в множество принятых к обработке, а также подавать как аргумент в процедуры алгоритма вместо перебираемых сущностей можно краткие перебираемые сущности, т. е. тройки (q_1, q_2, α), где $q_1, q_2 \in Q$; $\alpha \in A^*$. Модифицированный алгоритм не работает бесконечно долго и выдаёт исключение тогда и только тогда, когда его выдаёт исходный алгоритм. Более того, модифицированный алгоритм добавляет в очередь и в множество принятых к обработке столько же сущностей, сколько исходный алгоритм. Сложность модифицированного алгоритма по времени совпадает со сложностью исходного алгоритма, т. е. составляет $O(L|Q|^3|S|^4)$.

Поскольку как в очередь, так и в множество принятых к обработке добавляется не более чем $|S| + (2L-1)|Q|^3|S|^2$ сущностей, то в памяти хранится не более чем $2|S| + 2(2L-1)|Q|^3|S|^2 = O(L|Q|^3|S|^2)$ сущностей. Хранение одной краткой сущности требует $O(L \ln|Q| \ln|A|)$ бит памяти. Таким образом, сложность модифицированного алгоритма по памяти составляет $O(L^2|Q|^3|S|^2 \ln|Q| \ln|A|)$

Отметим, что особый интерес представляет частный случай, когда в структуре частичного стирания нет абсолютно различимых символов. В этом случае условие правильности упрощается: так как отсутствуют аномалии 1-го рода для функции F, то для принадлежности функции F классу правильных необходимо и достаточно отсутствия аномалий 2-го рода. В дальнейшем планируется публикация, посвящённая имеющему меньшую сложность алгоритму валидации, который приспособлен специально к указанному частному случаю.

ЛИТЕРАТУРА

1. *Казаков И. Б.* Передача информации в каналах, задаваемых структурами частичного стирания. Ч. 1 // Программная инженерия. 2020. Т. 11. № 5. С. 277–284.
2. *Казаков И. Б.* Передача информации в каналах, задаваемых структурами частичного стирания. Ч. 2 // Программная инженерия. 2020. Т. 11. № 6. С. 322–329.
3. *Казаков И. Б.* Критерий существования корректного протокола в канале частичного стирания // Чебышевский сборник. 2021. Т. 22. № 1. С. 133–151.
4. *Lampson B. W.* A note on the confinement problem // Comm. ACM. 1973. V. 16. No. 10. P. 613–615.
5. *McFarland J.* Covert Channels: An Overview. `https://www.researchgate.net/publication/330875417_Covert_Channels_An_Overview`. 2017.
6. *Wendzel S., Zander S., Fechner S., et al.* A pattern-based survey and categorization of network covert channel techniques // ACM Comput. Surveys. 2015. V. 47. No. 3. P. 1–26.
7. *Zander S., Armitage G., and Branch P.* Covert channels in multiplayer first person shooter online game // 33rd IEEE Conf. Local Computer Networks. Montreal, Quebec, Canada, 2008. P. 215–222.

REFERENCES

1. *Kazakov I. B.* Peredacha informatsii v kanalakh, zadavaemykh strukturami chastichnogo stiraniya. Ch. 1 [Transmission of information in channels specified by structures of partial erasure. P. 1]. Programmnaya Inzheneriya, 2020, vol. 11, no. 5, pp. 277–284. (in Russian)
2. *Kazakov I. B.* Peredacha informatsii v kanalakh, zadavayemykh strukturami chastichnogo stiraniya. Ch. 2 [Transmission of information in channels specified by structures of partial erasure. P. 2]. // Programmnaya Inzheneriya, 2020, vol. 11, no. 6, pp. 322–329. (in Russian)
3. *Kazakov I. B.* Kriteriy sushchestvovaniya korrektnogo protokola v kanale chastichnogo stiraniya [Criterion for the existence of a consistent protocol in a partial erasure channel]. Chebyshevckiy Sbornik, 2021, vol. 22, no. 1, pp. 133–151. (in Russian)
4. *Lampson B. W.* A note on the confinement problem. Comm. ACM, 1973, vol. 16, no. 10, pp. 613–615.
5. *McFarland J.* Covert Channels: An Overview. `https://www.researchgate.net/publication/330875417_Covert_Channels_An_Overview`. 2017.
6. *Wendzel S., Zander S., Fechner S., et al.* A pattern-based survey and categorization of network covert channel techniques. ACM Comput. Surveys, 2015, vol. 47, no. 3, pp. 1–26.
7. *Zander S., Armitage G., and Branch P.* Covert channels in multiplayer first person shooter online game. 33rd IEEE Conf. Local Computer Networks, Montreal, Quebec, Canada, 2008, pp. 215–222.