УДК 123.456.789
DOI: 10.55648/1998-6920-2022-16-2-12-22

# Применение специализированных детерминированных конечных автоматов для прогнозирования временных рядов

У. В. Павлова, А. А. Ракитский

В данной работе рассматривается возможность использования детерминированных конечных автоматов для прогнозирования временных рядов в режиме реального времени. Для решения этой проблемы авторами предлагается модификация автомата с десятью считывающими головками, предназначенного для распознавания мультилинейных последовательностей. В статье представлены модификации этого автомата, алгоритмы для их реализаций и приведены результаты, полученные с их помощью. Кроме того, в работе представлен метод изменения алгоритма работы автомата, обеспечивающий его пошаговое выполнение.

*Ключевые слова*: конечный автомат, прогнозирование, анализ временных рядов, теория информации, машинное обучение, сжатие данных.

## 1. Введение

В настоящее время исследования в области анализа и прогнозирования временных рядов охватывают практически все сферы деятельности человека: прогнозирование сбоев в работе различного оборудования; анализ экономических показателей; исследования различных данных в области медицины, таких как электрокардиограммы, электроэнцефалограммы и другие; природные явления, среди которых прогнозирование появления солнечных пятен, сейсмической активности, погодных явлений и многое другое. Все описанные выше примеры представляют собой временные ряды – последовательности данных, распределённых во времени (как правило, равномерно). В настоящее время большая часть данных записывается в режиме реального времени и имеет огромные объемы, в связи с чем необходимы надежные и эффективные средства для их обработки и анализа. К сожалению, большинство современных достаточно точных методов требуют огромных вычислительных ресурсов, поэтому разработка новых, высокопроизводительных, методов, позволяющих осуществлять анализ временных рядов в режиме онлайн, становится крайне актуальной.

В рамках представленной работы предлагается именно такой высокопроизводительный метод, позволяющий выявлять сложные паттерны во временных рядах в режиме реального времени. Предлагаемый метод базируется на конечном детерминированном автомате с десятью считывающими головками, который предназначен для распознавания мультилинейных последовательностей [11]. В работе исследуется возможность формирования таких конечных автоматов, которые смогли бы определять последовательность, выдаваемую неизвестным источником. Авторами рассмотрены несколько модификаций описанного выше автомата, позволяющих, во-первых, обрабатывать последовательности, отличающиеся от мультилинейных, а во-вторых, работать в режиме реального времени. Кроме того, рассмотрена возможность работы в сочетании с другими, уже известными, методами, для уточнения прогнозируемого значения. В последующих главах представлен обзор существующих методов прогнозирования,

описана модель автомата, представлены алгоритмы методов, предложенных авторами, а также результаты применения разработанного метода для некоторых реальных временных рядов.

## 2. Анализ предметной области

В работе [1] представлена модель, включающая в себя основные методы, используемые в машинном обучении, такие как искусственная нейронная сеть, многомерные адаптивные регрессионные сплайны, метод k-ближайших соседей и векторная регрессия опорных векторов. Авторы описывают применение разработанной модели к прогнозированию количества осадков в городах, а также приводят анализ эффективности своего алгоритма с входящими в его состав методами. В работе [2] описан метод пространственного прогнозирования электрической нагрузки с использованием модели клеточного автомата для пространственно-временного распределения новых нагрузок в определенной зоне обслуживания. Плотность электрической нагрузки для каждого из основных классов потребителей в каждой ячейке используется в качестве текущего состояния. Дополнительно устанавливается ряд правил обновления для имитации поведения роста и взаимодополняемости между классами. Положительные черты, выделяемые у этого метода, – хорошая производительность, небольшой объем необходимых данных и простота самого алгоритма, что позволяет масштабировать его без существенного увеличения сложности.

В [3] авторы описывают новый конечный обучающий автомат, который был разработан для снижения энергозатрат при преобразовании автоматов в детерминированные. Новый алгоритм содержит особый параметр для нахождения компромисса между точностью и потреблением энергии, что позволяет упростить машинное обучение. Благодаря своей детерминированности такой автомат может ускорить рабочий процесс и точно найти следующий символ в последовательности, используя небольшое количество операций, обеспечивая тем самым максимально быстрое прогнозирование.

Авторы статьи [4] разработали алгоритм Wayeb – инструмент, который пытается решить проблему прогнозирования сложных событий. Он использует автомат в вычислительной модели для обнаружения шаблонов и цепей Маркова, чтобы обеспечить вероятностное описание автомата. В статье [5] описаны регрессионные автоматы, которые предоставляют новый тип синтаксической модели для прогнозирования временных рядов. Построенный на основе обычных алгоритмов слияния состояний для идентификации автоматов, этот метод использует числовые данные в дополнение к символьным значениям и делает прогнозы на основе этих данных в режиме регрессии. Эти модели применяются для почасового прогнозирования скорости и энергии ветра. Также необходимо упомянуть возможность использования цепей Маркова [4, 6] и широко распространенного алгоритма ARIMA [7] для задач прогнозирования временных рядов.

В работах [8, 9] показано, что для прогнозирования временных рядов также могут быть использованы модели, основанные на использовании универсальных кодов (или методов сжатия данных). Так как архиваторы по своей структуре близки к автоматам и включают в себя алгоритмы сжатия данных, то их использование в задачах прогнозирования возможно как в качестве самостоятельного элемента, так и в комплексе с другими методами.

Конечные автоматы в анализе временных рядов – малоизученное направление. Возможность их применения только исследуется учеными. Конечный автомат способен не только распознавать слова, состоящие из символов теоретически бесконечного алфавита, но и при правильной реализации его можно обучить таким образом, чтобы он распознавал, к какому конкретно языку относится то или иное слово. В данной работе предлагается применение конечных автоматов для анализа и прогнозирования временных рядов, в которых могут присутствовать так называемые выбросы, т.е. отклонения от некоторого паттерна. К таким временным рядам, как правило, относятся реальные последовательности с неизвестными свойствами.

# 3. Теоретический базис

## 3.1. Задача прогнозирования

Прогнозирование является распространённой и востребованной задачей во многих областях человеческой деятельности. В результате использования методов прогнозирования уменьшается риск принятия неверных, необоснованных или субъективных решений. В целом задачу прогнозирования можно считать одной из самых сложных среди существующих задач. Она включает в себя множество неупорядоченных наборов данных, которые описывают поведение того или иного ряда. Для того чтобы составить прогноз, необходимо установить зависимость между всеми наборами, участвующими в прогнозе, а также проанализировать состояние каждого из них в прошлом и настоящем. Таким образом, можно сказать, что задача прогнозирования требует наличия некоторой выборки для обучения.

Очевидно, что данные, поступающие для обработки и дальнейшего прогнозирования, могут иметь сложную структуру, а их источник может быть недетерминированным или вероятностным и, кроме того, может быть чувствительным к внешним воздействиям. Если не накладывать на задачу какие-либо ограничения, упрощающие работу, то её можно отнести к разряду практически нерешаемых. В рамках работы будем считать, что источник данных полностью детерминирован, имеет конечный алфавит и генерирует не более одного символа в каждый момент времени. При таких упрощениях задача сводится к прогнозированию последовательности бесконечной длины.

## 3.2. Автоматы

Согласно определению, предложенному в [10], конечный автомат – абстрактная машина, число возможных внутренних состояний которой конечно.

Определим некоторый автомат $M$:

$$M = (V, Q, q_0, F, \delta),$$

где $V$ – входной алфавит (конечное множество входных символов), из которого формируются входные слова, воспринимаемые конечным автоматом;

$Q$ – множество внутренних состояний;

$q_0$ – начальное состояние ($q_0 \in Q$);

$F$ – множество заключительных, или конечных, состояний;

$\delta$ – функция переходов, значение которой есть множество всех состояний, в которые из данного состояния возможен переход по конкретному входному символу или пустой цепочке ($\varepsilon$).

Конечный автомат начинает работу в состоянии $q_0$, последовательно считывая по одному символу входного слова либо подцепочки символов этого слова (которые также можно рассмотреть как один символ, если провести преобразование алфавита). После прочтения каждого символа автомат переходит в новое состояние в соответствии с функцией переходов.

В рассматриваемой модели каждый автомат $M$ принимает на вход бесконечное слово $\alpha$ и выдаёт на выходе бесконечное слово $M(\alpha)$ с таким ограничением, что, прежде чем считать очередной $i$-й символ, автомат должен вывести $M(i)$, то есть спрогнозировать следующий символ в слове. Если каждый $i$-й символ $M(\alpha)$ равен $i$-му символу $\alpha$, то можно сказать, что автомат $M$ полностью освоил цепочку $\alpha$. В случае, когда некоторый автомат $M$ сможет полностью распознать последовательность $\alpha(A)$ – слово над $A$, где $A$ – конечный алфавит, то можно утверждать, что автомат полностью освоил эту последовательность.

В дальнейшем автомату $M$ подается на вход бесконечное слово $\alpha$. На основе первого считанного символа и в соответствии со своим текущим состоянием автомат начинает делать пе-

реходы. При каждом движении он перемещает считывающую головку вправо и делает предположение о том, каким будет следующий символ во входящей последовательности. Общая схема работы детерминированного конечного автомата представлена на рис. 1.

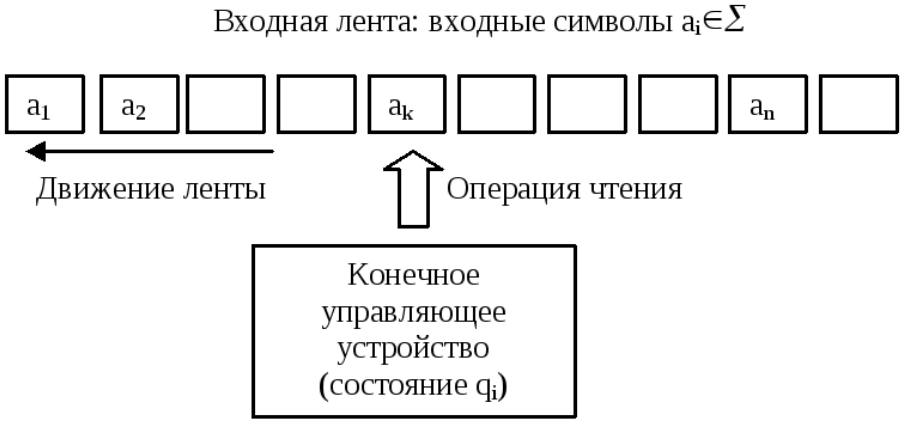


Рис. 1. Детерминированный конечный автомат

### 3.3. Автомат для распознавания мультилинейных последовательностей

Оригинальный автомат был представлен автором в виде псевдокода в работе [11], что оставляет пространство для различных его реализаций. Кратко опишем основные принципы работы алгоритма.

Любую бесконечную мультилинейную последовательность можно представить в виде:

$$q \prod_{n \geq 1} \prod_{i \geq 1}^{m} p_i s_i^n \tag{1}$$

для некоторых $m \geq 1$ и строк $q$, $p_i$, $s_i$, разделенных на блоки типа $p_i s_i^n$. То есть некоторая последовательность $\alpha$ может быть разбита на блоки, которые состоят из сегментов типа $p_i s_i^n$ и начального блока $q$. Для того чтобы полностью освоить $\alpha$, автомат $M$ будет чередовать в своей работе две процедуры – *Коррекция* и *Сопоставление*. *Коррекция* пытается расставить головки $h_1$, $h_2$, $h_3$, $h_4$ в начало каждого сегмента. Более того, $h_2$ должна находиться после $h_1$ на заданном количестве сегментов (длина сегментов может отличаться), $h_3$ должна находиться после $h_2$ на таком же количестве сегментов и $h_4$ должна находиться после $h_3$ на таком же количестве сегментов. Каждый раз, когда запускается *Коррекция*, количество символов между головками увеличивается на 1. *Сопоставление* пытается освоить $\alpha$, предполагая, что *Коррекция* была успешной и количество сегментов, разделяющих головки, одинаково для каждой из них. *Сопоставление* работает, используя $h_1$, $h_2$, и $h_3$ для прогнозирования $h_4$. Более «поздние» головки обозревают бо́льшее количество копий сегментов $p_i s_i^n$, то есть $h_4$ обозревает больше копий, чем $h_3$, которая обозревает больше копий, чем $h_2$, которая обозревает больше копий, чем $h_1$. *Сопоставление* сдвигает все головки одновременно, сравнивая их между собой и определяя, в какой момент каждая из головок достигла конца своего сегмента. Как только все головки достигнут конца своих сегментов, *Сопоставление* начинается заново. Если обнаружена какая-либо проблема, *Сопоставление* завершается, и *Коррекция* запускается снова.

На рис. 2 показаны первые три шага процедуры *Коррекция*. Здесь видно, как автомат пытается расположить все головки таким образом, чтобы существующая последовательность разбивалась на блоки.

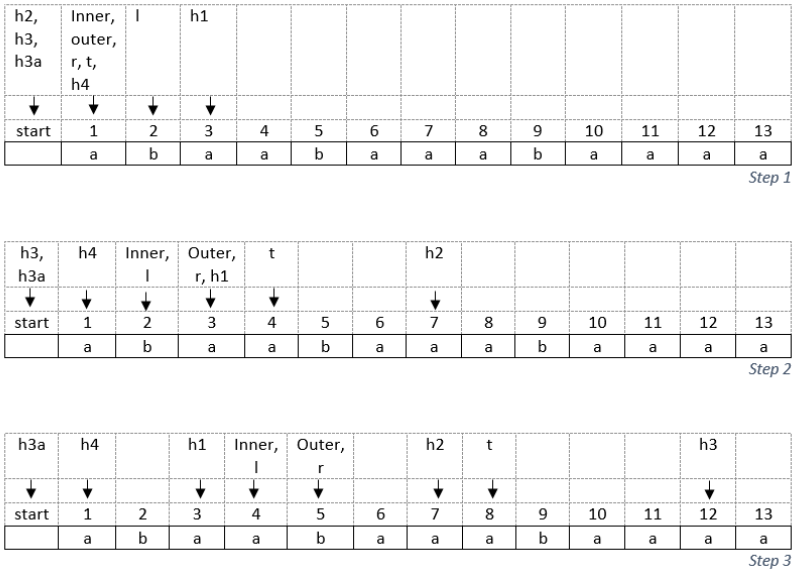


Рис. 2. Схема работы процедуры *Коррекция*

На рис. 3 показана работа процедуры *Сопоставления* после завершения *Коррекции*. Все считывающие головки расположены в начале блоков и готовы к работе. Первый шаг показывает начальное положение головок, а второй шаг показывает положение после перехода к следующему символу. Как только первая головка ($h_1$) достигает нового символа «b», *Сопоставление* завершается.

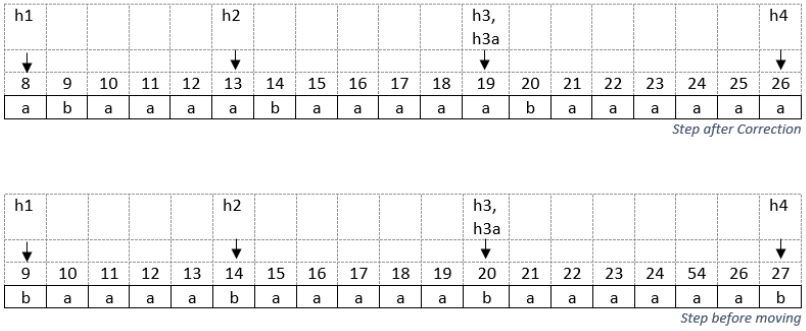


Рис. 3. Схема работы процедуры *Сопоставление*

## 4. Алгоритм пошагового исполнения

В процессе исследования была разработана модель [12], максимально похожая на описанную ранее. К ней было применено несколько модификаций, подробно описанных в работах [13, 14], а также интегрирована работа с архиваторами [15]. Результаты тестирования, приведенные в работах, коренным образом не меняют результаты, полученные при первичном тестировании исходной модели автомата. Таким образом, принято решение об изменении конструкции и принципа работы автомата, но при этом сохранении логики его работы.

В ходе предыдущей работы над моделями автомата возникала проблема невозможности остановки алгоритма в конкретный момент времени и внесения изменений в его работу. Новый алгоритм реализован с возможностью модели останавливаться и запоминать свое положение каждый раз после того, как крайняя правая головка была перемещена и был сделан прогноз, а также начинать работу с места последнего «сохранения», а не с начала последовательности, как это предполагается в исходной версии автомата.

Поскольку принципы машинного обучения должны быть соблюдены, необходимо разделить входную последовательность на обучающую и тестовую выборки. Из этого следует, что

и работа автомата должна быть разграничена на зоны ответственности по обработке последовательности. В представленном алгоритме было введено два режима – обучения и генерации. Ниже каждый из них кратко описан с теоретической точки зрения.

В режиме обучения из последовательности, поступившей автомату в качестве входных данных, выделяется и сохраняется паттерн, исходя из которого будет проводиться дальнейшая работа алгоритма. Стоит отметить, что размер обучающей выборки ограничен определенным количеством символов и не может превышать заданного фиксированного значения. После успешного завершения режима обучения запускается режим генерации. В этой ветке работы автомат, исходя из выбранного ранее шаблона, определяет следующий символ в последовательности. Далее подробно рассмотрим автомат и его работу на примере псевдокода.

```
Input: Sequence, data_size, learn_size
Output: Automaton result
   create multiHeadAutomaton  A (sequence, data_size)
   If (A->learn(learn_size)) success:
       A->generateNext (generate_size)
```

Алгоритм 1. Main-функция

В main-функции программы (см. Алгоритм 1) на вход поступает последовательность, с которой будет происходить дальнейшая работа, её размер и длина обучающей выборки. В процессе создания экземпляра класса конструктор (Алгоритм 2) принимает непосредственно строку с данными и её длину. После создания по отношению к созданному экземпляру запускается процесс обучения, а в случае его успешного завершения – процесс генерации.

```
class multiHeadAutomaton:
   data_type data
   vector <data_type> predict_arr
   data_size, match_state, h1, h2, h3, h3a, h4, t, l, r, inner, outer, corr
   pattern_check = 0
   multiHeadAutomaton(data_type *d, int dsize):data(d),data_size(dsize)
```

Алгоритм 2. Поля и конструктор класса

Перейдем к более детальному описанию класса автомата. Полями являются считывающие ($h_1$, $h_2$, $h_3$, $h_4$) и вспомогательные (остальные) головки автомата, с помощью которых происходит основная работа, а также дополнительные переменные, необходимые в процессе выполнения алгоритма. Кроме того, как показано в Алгоритме 1, класс содержит конструктор и несколько методов, которые необходимы для его работы. В рамках данной работы подробно будут рассмотрены основные из них. Подробное описание остальных методов можно найти в работах [11, 12].

Прежде чем запустить один из режимов, автомат считывает входную последовательность и размер её обучающей выборки. Далее запускается непосредственно режим обучения. Он начинается с вызова метода *learn()* (см. Алгоритм 3), на вход которому поступает размер обучающей выборки. Внутри себя функция вызывает два ключевых метода – *Коррекцию* и *Сопоставление*.

Метод *Коррекция* сохранил свой первоначальный вид и не претерпел изменений в структуре. Внутри него автомат расставляет считывающие головки по последовательности и с помощью них, а также вспомогательных выделяет паттерн. Если паттерн выделился успешно, то он проверяется в процедуре *Сопоставления*. В том случае, если вернулся 0, паттерн считается ошибочным, и *Коррекция* запускается снова. Если шаблон не был выделен, то размер обучающей выборки был слишком мал либо последовательность содержит много стохастических выбросов. В таком случае за паттерн принимается последняя найденная подстрока.

```
Input: lsize
Output: Learn result
  while h4 < lsize:
    Correction()
    while Matching(lsize)==0
```

Алгоритм 3. Функция *learn()*

После завершения функции *Коррекция* вызывается функция *Сопоставление* (Алгоритм 4), в которой происходит основная работа и проверяется правильность определенного паттерна. Автомат сравнивает каждый символ выбранного шаблона на равенство соответствующему символу в обрабатываемой последовательности.

```
Input: Automaton state (h1, h2, h3, h3a, h4, r, l, t, inner, outer), size
Output: Matching result
Loop:
If match_state <= 0:
    h3a := h3
    match_state := 0
  If match_state <= 1:
    If h4 > size:
return 1
    match_state := 1
    If step_one():
      return 0
    If data[h2] <> data[h4]:
      return 1
  …
  If match_state <= 4:
    If h4 > size:
      return 1
    match_state := 4
    If step_four():
      return 0
    match_state := 0
    If data[h3a] <> data[h4]:
      data[h4] := data[h3a]
      h4 + 1
      h3a + 1
End loop
```

Алгоритм 4. Функция *Сопоставление*

Сначала проверяется текущее состояние автомата. В момент, когда автомат определил свое фактическое состояние, в зависимости от его значения он запускает одну из внутренних функций *step()* (Алгоритм 5). Эти функции перемещают считывающие головки и вызывают функцию прогнозирования. В ней автомат сравнивает предсказанный символ с символом, содержащимся в исходной последовательности. Если символы равны, это означает, что нужно сохранить символ и рассматривать его как часть паттерна. В противном случае найденный паттерн неверен, поэтому автомат прекращает *Сопоставление* и снова запускает *Коррекцию*.

```
Input: Automaton state
Output: Step result
While (data[h1] = data[h2]) and (data[h1] = data[h3]) and (data[h1] = data[h4]):
    h1 + 1
    h2 + 1
```

```
    h3a + 1
    h3 + 1
    h4 + 1
    Guess(h2, h4)
```

Алгоритм 5. Функция *step()*

Если правильный паттерн был успешно выделен, автомат должен прекратить вызывать функцию *Коррекции*. Когда это событие произошло, мы можем завершить обучение и запустить режим генерации, в котором обязательно необходимо указать, сколько символов последовательности нужно сгенерировать. В противном случае автомат сгенерирует один символ и прекратит работу. В результате внесенных изменений, представленных выше, мы получили автомат, полностью соответствующий исходному по логике выполнения.

## 5. Результаты тестирования

Для того чтобы проанализировать работу автомата, был проведен ряд тестов с использованием различных последовательностей. В данной работе рассматривается временной ряд с алфавитом {0, 1}, поэтому для оценки точности прогноза не принимается во внимание математическое ожидание и дисперсия, но ведется подсчет ошибок, совершенных автоматом во время работы. Под верным прогнозом будем понимать значение, которое спрогнозировал автомат, совпадающее со значением в исходной последовательности.

Подробно о предыдущих результатах говорилось в работах [12, 15]. Для дополнительного исследования возможностей автомата было проведено тестирование на псевдомультилинейной последовательности [16]. В исходном виде данная строка не имеет паттерна, поэтому автомат, как и ожидалось, не смог обработать строку и совершить какой-либо прогноз. Очевидно, что при данном результате последовательность должна подвергнуться предварительной обработке. В её начало вручную был добавлен паттерн. График обработанной последовательности отображен на рис. 4. На графике можно увидеть, каким образом изменяется последовательность в зависимости от количества считанных данных.

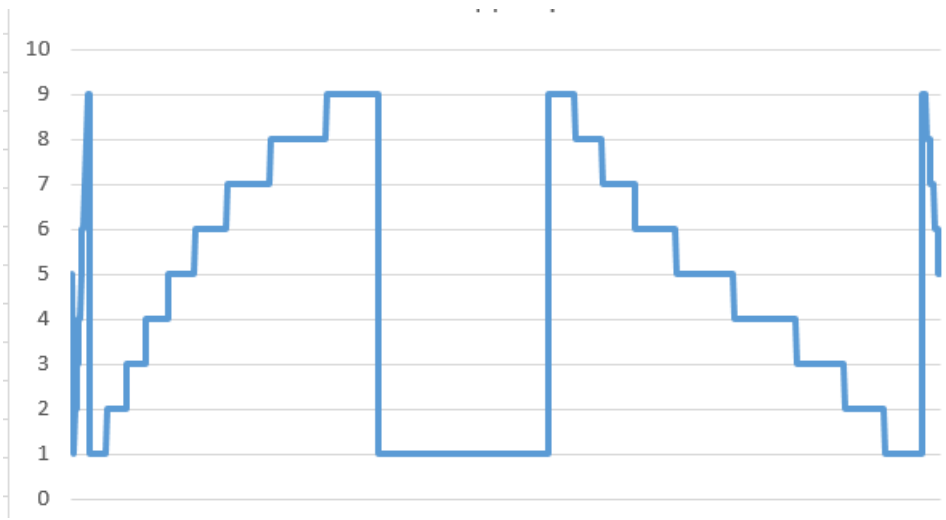


Рис. 4. Псевдомультилинейная последовательность

В ходе тестирования длина паттерна несколько раз была изменена. Это дало возможность оценить работу автомата при избыточной и недостаточной длине паттерна. Результаты приведены в табл. 1.

Таблица 1. Результаты тестирования

| Длина последовательности | Размер обучающей выборки | Процент верных прогнозов |
|---|---|---|
| 1000000 | 100000 | 86 % |
| 1000000 | 10000 | 32 % |
| 1000000 | 300000 | 65 % |

Длина последовательности оставалась неизменной на протяжении всего тестирования. Первоначальный размер паттерна равнялся 10 % длины строки. В режиме обучения автомат выделил закономерность и перешел в режим генерации. Сравнение сгенерированной и первоначальной строки показало, что точность сделанных прогнозов достаточно высока для последовательности, не являющейся мультилинейной.

Далее размер обучающей выборки был сильно сокращен, что повлекло за собой ухудшение качества прогноза. Это объясняется особенностями работы автомата – в режиме обучения он успевает полностью выделить паттерн, что вызывает дальнейшие несовпадения при сравнении сгенерированной последовательности с оригинальной.

При увеличении длины паттерна процент верно сделанных предположений снижается, хоть и некритично. Автомат полностью выделяет не только сам паттерн, но и часть последовательности, что и вносит погрешности в дальнейшую работу.

## 6. Заключение

В статье рассматривается возможность применения детерминированных конечных автоматов в задаче прогнозирования временных рядов. Был рассмотрен десятиголовочный автомат, настроенный на работу с мультилинейными последовательностями, а также описаны его недостатки при работе с временными рядами, отклоняющимися от паттерна. Основное внимание уделено совершенно новому алгоритму, способному работать в пошаговом режиме. Алгоритм включает несколько режимов работы – обучение и прогнозирование. Каждый из этих режимов выполняет определённую функцию: обучение настраивает автомат и выделяет паттерн, а прогнозирование генерирует последовательность и сравнивает ее с исходной. Приведены результаты тестирования разработанного метода на последовательностях, близких к мультилинейным.

## Литература

1. *Sumi S. M., Zaman M. F., Hirose H.* A rainfall forecasting method using machine learning models and its application to the Fukuoka city case // International Journal of Applied Mathematics and Computer Science. 2012. V. 22, № 4. P. 841–854.
2. *Carreno E. M., Rocha R. M., Padilha-Feltrin A.* A cellular automaton approach to spatial electric load forecasting // IEEE Transactions on Power Systems. 2010. V. 26, №. 2. P. 532–540.
3. *Abeyrathna K. D.* A Novel Multi-step Finite-State Automaton for Arbitrarily Deterministic Tsetlin Machine Learning // International Conference on Innovative Techniques and Applications of Artificial Intelligence. Cambridge, UK, December 15–17, 2020. V. 12498. P. 108.
4. *Alevizos E., Artikis A., Paliouras G.* Wayeb: a tool for complex event forecasting // 22nd International Conference on Logic for Programming, Artificial Intelligence and Reasoning. Awassa, Ethiopia, November 16–21, 2018.
5. *Qin L., Hammerschmidt C., Pellegrino G., Verwer S*. Short-term Time Series Forecasting with Regression Automata // ACM SIGKDD 2016 Workshop on Mining and Learning from Time Series (MiLeTS). San Francisco, California, USA, August 13–17, 2016. DOI: 10.1145/1235.

6. *Alevizos E., Artikis A., Paliouras G.* Event Forecasting with Pattern Markov Chains // 11th ACM Int. Conf. on Distributed and Event-based Systems, 2017. P. 146–157.
7. *Jenkins G. M., Box G.* Time Series Analyses. Forecasting and control. Holden-Day, 1976. 589 p.
8. *Lysyak A. S., Ryabko B. Y.* Time series prediction based on data compression methods // Problems of Information Transmission. 2016. V. 52, № 1. P. 92–99.
9. *Чирихин К. С., Рябко Б. Я.* Экспериментальное исследование точности методов прогноза, базирующихся на архиваторах // Вестник Новосибирского государственного университета. Серия: Информационные технологии. 2018. Т. 16, №. 3. С. 145–168.
10. *Hopcroft J. E., Ullman J. D.* Introduction to automata theory, languages, computation. Addison-Wesley publishing company, 1979. 427 p.
11. *Smith T.* Prediction of infinite words with automata // Theory of Computing Systems. 2018. V. 62, № 3. P. 653–681.
12. *Павлова У. В., Ракитский А. А.* Исследование возможности применения автоматов для прогнозирования временных рядов // РНТК «Обработка информации и математическое моделирование», СибГУТИ, 25–26 апреля 2019. С. 168–172.
13. *Pavlova U., Rakitskiy A.* Time Series Forecasting Method Based on Finite State Machine // International Conference of Young Specialists on Micro/Nanotechnologies and Electron Devices (EDM). Aya, Altai Region, Russia, June 30 – July 3, 2021. P. 533–536.
14. *Pavlova U., Rakitskiy A.* Development and Research of the Time Series Prediction Method Based on Finite State Automaton // Ural Symposium on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT), Yekaterinburg, Russia, 13–14 May, 2021. P. 305–307.
15. *Павлова У. В., Ракитский А. А.* Разработка и исследование метода прогнозирования временных рядов на основе конечных автоматов // РНТК «Обработка информации и математическое моделирование», СибГУТИ, 23–24 апреля 2020. С. 142–145.
16. UCI Machine Learning Repository [Электронный ресурс]. URL: `https://archive.ics.uci.edu/ml/index.php` (дата обращения: 28.01.2022).

*Статья поступила в редакцию 18.03.2022;*
*переработанный вариант – 05.04.2022.*

**Павлова Ульяна Владимировна**
аспирант 1-го курса института ИВТ СибГУТИ, ассистент кафедры ПМиК СибГУТИ (630102, Новосибирск, ул. Кирова, 86), e-mail: `uljana.pavlova2012@yandex.ru`.

**Ракитский Антон Андреевич**
к.т.н, доцент кафедры ПМиК (630102, Новосибирск, ул. Кирова, 86); научный сотрудник НГУ; старший научный сотрудник ИВТ СО РАН, e-mail: `rakitsky.anton@gmail.com`.

**Application of specialized DFA for time series forecasting**

**Ulyana V. Pavlova**
Postgraduate student, Department of Appl. Math. and Cybernetics, Siberian State University of Telecommunications and Information Science (SibSUTIS, Novosibirsk, Russia), `uljana.pavlova2012@yandex.ru`.

**Anton A. Rakitskiy**
Candidate of technical sciences, Docent, Siberian State University of Telecommunications and Information Science (SibSUTIS, Novosibirsk, Russia), `rakitsky.anton@gmail.com`.

In this paper the possibility of using deterministic finite automaton to forecast time series in real time is considered. To solve this problem, a modification of the automaton with 10 reading heads designed to recognize multilinear sequences is proposed. The article presents modifications of this automaton, algorithms for their implementation, and demonstrates the results for various time series. In addition, the paper presents a method for changing the algorithm of the automaton ensuring its step-by-step execution.

*Keywords*: finite state automaton, forecasting, time series analysis, information theory, machine learning, data compression.

**References**

1. Sumi S. M., Zaman M. F., Hirose H. A rainfall forecasting method using machine learning models and its application to the Fukuoka city case. *International Journal of Applied Mathematics and Computer Science*, 2012, vol. 22, no. 4, pp. 841-854. DOI: 10.2478/v10006-012-0062-1.
2. Carreno E. M., Rocha R. M., Padilha-Feltrin A. A cellular automaton approach to spatial electric load forecasting. *IEEE Transactions on Power Systems,* 2010, vol. 26, no. 2, pp. 532-540. DOI: 10.1109/tpwrs.2010.2061877.
3. Abeyrathna K. D. A Novel Multi-step Finite-State Automaton for Arbitrarily Deterministic Tsetlin Machine Learning. *International Conference on Innovative Techniques and Applications of Artificial Intelligence*, Cambridge, UK, 15-17 December, 2020, vol. 12498, pp. 108. DOI: 10.1007/978-3-030-63799-6_8.
4. Alevizos E., Artikis A., Paliouras G. Wayeb: a tool for complex event forecasting. *22nd International Conference on Logic for Programming, Artificial Intelligence and Reasoning*, Awassa, Ethiopia, 16-21 November, 2018.
5. Qin L., Hammerschmidt C., Pellegrino G., Verwer S. Short-term Time Series Forecasting with Regression Automata. *ACM SIGKDD 2016 Workshop on Mining and Learning from Time Series (MiLeTS)*, San Francisco, California, USA, 13-17 August, 2016. DOI: 10.1145/1235.
6. Alevizos E., Artikis A., Paliouras G. Event Forecasting with Pattern Markov Chains. *11th ACM Int'l Conf. on Distributed and Event-based Systems*, 2017, pp. 146-157. DOI: 10.1145/3093742.3093920.
7. Jenkins G. M., Box G. *Time Series Analyses. Forecasting and control*. Holden-Day, 1976, 589 p.
8. Lysyak A. S., Ryabko B. Y. Time series prediction based on data compression methods. *Problems of Information Transmission*, 2016, vol. 52, no. 1, pp. 92-99. DOI: 10.1134/s0032946016010075.
9. Chirikhin, K. S., and B. Ya. Ryabko. Jeksperimental'noe issledovanie tochnosti metodov prognoza, bazirujushhihsja na arhiva-torah [Experimental Study of the Accuracy of Compression-Based Forecasting Methods]. *Vestnik NSU, Series: Information Technologies*, 2018, vol. 16, no. 3, pp. 145-158. DOI: 10.25205/1818-7900-2018-16-3-145-158.
10. Hopcroft J.E., Ullman J.D. *Introduction to automata theory, languages, computation*. Addison-Wesley publishing company, 1979, 427 p.
11. Smith T. Prediction of infinite words with automata. *Theory of Computing Systems*, 2018, vol. 62, no. 3, pp. 653 - 681. DOI: 10.1007/s00224-016-9739-4.
12. Pavlova U. V., Rakitsky A. A. Issledovanie vozmozhnosti primenenija avtomatov dlja prognozirovanija vremennyh rjadov [Investigation of the possibility of using automata to predict time series]. *Rossijskaja nauchno-tehnicheskaja konferencija*, Novosibirsk, 25-26 April, 2019, pp. 168-172.
13. Pavlova, U., Rakitskiy, A. Time Series Forecasting Method Based on Finite State Machine. *2021 IEEE 22nd International Conference of Young Professionals in Electron Devices and Materials (EDM)*, Aya, Altai Region, Russian Federation, 30 June – 3 July, 2021, pp. 533-536. DOI: 10.1109/edm52169.2021.9507729.
14. Pavlova, U., Rakitskiy, A. Development and Research of the Time Series Prediction Method Based on Finite State Automaton. *2021 Ural Symposium on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT)*, Yekaterinburg, Russian Federation, 13-14 May, 2021, pp. 305-307. DOI: 10.1109/usbereit51232.2021.9455056.
15. Pavlova U. V., Rakitsky A. A. Razrabotka i issledovanie metoda prognozirovanija vremennyh rjadov na osnove konechnyh avtomatov [Development and research of a time series forecasting method based on finite automata]. *Rossijskaja nauchno-tehnicheskaja konferencija*, Novosibirsk, 23-24 April, 2020, pp. 142-145.
16. UCI Machine Learning Repository, available at: `https://archive.ics.uci.edu/ml/index.php` (accessed 28.01.2022).