Санкт-Петербургский государственный университет информационных технологий, механики и оптики

Кафедра «Компьютерные технологии»

Д. И. Суясов

Разработка алгоритмов распознавания текста на основе клеточных автоматов

Магистерская диссертация

Руководитель – А.А. Шалыто

Санкт-Петербург
2007

Оглавление

ВВЕДЕНИЕ

Вычислительные системы существуют уже не один десяток лет. Целью их создания являлась возможность заменить человека, сделать за него трудоемкую работу, требующую сложных вычислений.

Одна из таких задач состоит в том, как научить компьютер распознавать образы (в частности, какой алгоритм необходимо создать, чтобы компьютер мог воспринимать изображения текстов).

Распознавание текстов – очень сложная задача с теоретической и практической точек зрения. Человек, например, задействует для этого весь комплекс знаний и опыта. Он определяет текст из совокупности сигналов органов чувств, выделяет каждый символ, выделяет характерные признаки символов и на основании своего опыта приходит к выводу о значении символа и всего текста в целом.

Компьютер ошибается в процессе распознавания намного чаще человека. Сегодня не существует абсолютно точного метода определения текста и символа по их изображению. Многие разработанные коммерческие проекты используют свои запатентованные методы и не могут похвастаться идеальным решением задачи.

Во многих случаях хорошее решение дает комплексный подход к задаче. Сама задача распознавания текста разделяется на подзадачи: фильтрация изображения от шума, выделение изображений символов из изображения текста, выделение признаков символов и сравнение этих признаков с сохраненными образцами. Каждая задача содержит много вариантов решений, из которых лишь некоторые являются более или менее оптимальными.

Достаточно развитым направлением науки на сегодняшний день являются клеточные автоматы [1, 2]. Особый интерес к клеточным автоматам

проявляется прежде всего благодаря их простоте: на основе простых правил клеточные автоматы могут порождать сложное поведение. Кроме того, клеточные автоматы – это идеальный вариант для параллельных вычислений, они могут эффективно использоваться в многопроцессорных системах или быть реализованы аппаратно, так как основное свойство их правил – локальность и однородность.

Преимущества клеточных автоматов могут оказаться полезными в системе распознавания текста. Простота и однородность правил может позволить создавать сложные системы на базе нескольких логических или математических элементов и добиться результата с меньшими затратами как вычислительных ресурсов, так и памяти.

Основным направлением исследований, которые проводятся в работе, является определение границ применимости клеточных автоматов в задачах и подзадачах, возникающих в процессе распознавания текста. Для этого необходимо выполнить анализ клеточных автоматов и их свойств, выделить основные характеристики клеточных автоматов, необходимых в решении задач распознавания текста, и разработать соответствующие алгоритмы.

В процессе исследования необходимо создать модель и программу на ее основе для реализации разработанных идей и алгоритмов.

ГЛАВА 1. ПОСТАНОВКА ЗАДАЧИ

1.1. Распознавание текста

Распознавание текста является одним из направлений распознавания образов [3]. Распознавание образов представляет собой очень сложную задачу в теоретическом и практическом смыслах, несмотря на то, что с ней достаточно легко справляются многие живые организмы и человек. Крайне сложно создать искусственную систему и ее технически реализовать для того, чтобы эффективно выполнять данный процесс. В данном случае, под распознаванием понимается соотнесение изображения объекта, его образа, набора признаков самому объекту.

Примерами и приложениями систем распознавания образов могут являться как распознавание текста в общем, так и отдельных его символов, распознавание речи, человеческих лиц, биометрических данных человека, штрих-кодов продуктов, номеров машин и т.д.

Примерами распознавания текста являются: оцифровка изображений текста (сканированные книги, статьи, журналы) для последующей работы с его цифровым аналогом, обработка анкетных бланков, распознавание номеров машин и надписей на объектах и т.д. Задача распознавания текста остается актуальной на сегодняшний день, так как не существует сто процентной универсальной системы по распознаванию текста [4].

Система распознавания текста предполагает наличие на входе изображения с текстом (в формате данных графического файла). На выходе системы должен сформироваться текст, выделенный из этого изображения.

Распознавание текста включает в себя следующие подзадачи и подпроцессы.

1. Поступающее на вход системы изображение должно быть очищено от шума и приведено к виду, позволяющему эффективно выделять символы и распознавать их.

2. Система должна разбить изображение на блоки текста, основываясь на особенностях его выравнивания и распределения по нескольким колонкам.

3. Изображение с текстом должно быть разделено на изображения строк, а затем на изображения символов для того, чтобы в дальнейшем обработать каждый символ по отдельности.

После данного шага разные системы распознавания работают по своим специфическим алгоритмам.

4. Изображение символа может обрабатываться целиком, для этого оно сравнивается с имеющимися шаблонами. Другим вариантом является выделение характеристик изображаемого символа: отбор характерных признаков, и классификация данных признаков по имеющимся в системе критериям.

На выходе четвертого шага появляется возможный вариант буквы. Однако обычно системы на этом не останавливаются и продолжают работу на основе других методов, уточняя полученный результат.

5. Результат распознавания может быть не удовлетворительным. Для получения более хороших результатов в системе может быть встроен блок обучения. С помощью этого блока можно задать системе примеры начертания разных букв в данном шрифте. После процесса обучения предполагается лучшее качество распознавания текста.

Система распознавания текста не всегда должна следовать всем описанным шагам, но основные действия процесса распознавания являются общими для любого алгоритма.

1.2. Существующие решения

Существует несколько основных систем распознавания текста. Все они являются коммерческими продуктами, и многие внутренние алгоритмы их работы скрыты от общего доступа. Принцип работы подобных систем базируется на нескольких стратегиях [4], но, зачастую, алгоритм распознавания в самом общем виде состоит в последовательном выдвижении и проверке гипотез. При этом порядок их выдвижения управляется заложенными в программу знаниями об исследуемом предмете и результатами проверки предыдущих гипотез.

1.2.1. Распознавание по шаблонам

Программное обеспечение *OCR* (*Optical Character Recognition* – оптическое распознавание символов) обычно работает с большим растровым изображением страницы из сканера. При этом большинство систем имеет шаблоны, созданные для различных начертаний. После нескольких распознанных слов, программное обеспечение определяет используемый шрифт и ищет соответствующие пары только для этого шрифта. В некоторых случаях программное обеспечение использует численные значения частей символа (пропорций), чтобы определить новый шрифт. Это может улучшить эффективность распознавания.

Программа распознавания *TypeReader* использует машинно-зависимые алгоритмы на основе шаблонного подхода.

Данный подход требует создания шаблона для каждого шрифта. Например, программа *TypeReader* использует 2100 различных вариантов начертаний символов.

1.2.2. Структурный подход

Самая продаваемая в мире система *OCR – Caere OmniPage Professional* использует алгоритм, основанный на нахождении общих специфических особенностей символов.

Эта система содержит 100 различных алгоритмов для идентификации 100 различных символов: верхнего и нижнего регистра от «A» до «Z», записи чисел и символов пунктуации. Каждый из этих алгоритмов ищет «особенности» начертаний типа «островов», «полуостровов», точек, прямых оттисков и дуг. Экспертные системы также рассматривают горизонтальные и вертикальные проекции оттисков буквы и обращают внимание на основные особенности в созданных кривых, суммируя в них число темных пикселей.

К сожалению, нечеткий текст может стать специфической проблемой для этих структурных алгоритмов, так как отсутствующий пиксель может разбивать длинный штрих или кривую, а дополнительное пятно грязи может закрывать петлю.

1.2.3. Контекстное распознавание

В программное обеспечение системы *OCR* часто включается словари для помощи алгоритмам распознавания. Словари предоставляют справки во многих случаях, но быстро отказывают, когда, например, имеют дело с именами собственными, которые не находятся в словаре. Этот эффект особенно заметен в российской программе *FineReader*, который чаще, чем в среднем по всем символам, ошибается в словах, которые отсутствует в его словаре.

1.2.4. Нейронные сети в системе распознавания

Нейронные сети – это структура связанных элементов, на которых заданы функции преобразования сигнала, а также коэффициенты, которые могут быть настроены на определенный характер работы. Часть элементов

структуры выделены как входные: на них поступают сигналы извне, часть – выходные: они формируют результирующие сигналы. Сигнал, который проходит через нейронную сеть, преобразуется согласно формулам на элементах сети, и на выходе формируется ответ [5, 6].

Нейронная сеть может служить в системе распознавания текста в качестве классификатора. Этот классификатор можно обучать, настраивая коэффициенты на элементах сети, и, таким образом, стремиться к идеальному результату распознавания.

Нейронные сети с успехом могут применяться в системах распознавания текста, но существует большое число недостатков, которые препятствуют их широкому применению. Для построения сети, обеспечивающей распознавание каждого символа текста, необходимо построить достаточно большую сеть элементов, что приводит к большим затратам памяти. Еще сильнее тратятся ресурсы системы в процессе распознавания, так как функции на элементах сети работают с числами с плавающей точкой. Кроме этого нейронные сети необходимо обучать на все случаи, что, однако, не гарантирует точного результата. И, наконец, работа нейронной сети по распознаванию текста во многом зависит от конфигурации сети и функций, заданных в элементах, что требует больших усилий для построения эффективно работающей сети.

1.2.5. Решение задачи распознавания текста на базе клеточных автоматов

Попытки решения задачи распознавания предпринимались и на основе клеточных автоматов [7, 8]. К сожалению, дальше высказываний о возможности построения клеточного автомата для распознавания текста, в данных работах речи не идет.

1.2.6. Другие варианты решения задачи распознавания текста

Распознавание текста – это комплекс задач, которые необходимо выполнить для получения конечного результата (текста). Действующие коммерческие системы распознавания текста пользуются набором алгоритмов, которые в совокупности дают весьма точный результат.

Одна часть системы распознавания может функционировать на основе нейронной сети, другая часть может использовать преимущества клеточных автоматов, и, наконец, третья часть системы – накапливать статистику и на ее основе выдавать результат.

Комплекс мер и алгоритмов, бесспорно, позволит добить большего результата, нежели отдельно взятый принцип или алгоритм.

Клеточные автоматы могут быть частью подобного комплекса. Они имеют бесспорные преимущества, такие как возможность параллельного вычисления, легкость и простота правил, на основе которых они построены, возможность реализации многих сложных алгоритмов обработки изображений.

1.3. Формальная постановка задачи

Необходимо исследовать процесс распознавания текста, выделить основные составляющие данного процесса. Следует определить свойства и принципы, на основе которых можно использовать клеточные автоматы в данном процессе. При этом необходимо:

- исследовать процесс обработки изображения на основе клеточных автоматов;
- разработать алгоритм разбиения изображения текста на изображения символов с использованием клеточных автоматов;

- определить принцип выделения признаков символов на основе клеточных автоматов;
- разработать алгоритм выделения признаков символов;
- определить возможность построения клеточного автомата в процессе классификации признаков символов.

ГЛАВА 2. ПОСЛЕДОВАТЕЛЬНОСТЬ КЛЕТОЧНЫХ АВТОМАТОВ

2.1. Понятие клеточного автомата

Понятие клеточного автомата не является новым. Оно описано во многих книгах [1, 9] и, более того, есть люди, которые посвятили клеточным автоматам всю жизнь [2].

История клеточных автоматов начинается более полувека назад. Многие книги указывают, что клеточный автомат – это результат работы Джона фон Немана (John von Neumann) и Конрада Цузе (Konrad Zuse), которые в начале сороковых годов 20 века разработали в идею универсальной вычислительной среды для построения, анализа и сравнения характеристик алгоритмов [10]. Работы этих ученых базируются на исследованиях Станислава Улама (Stanislaw Ulam) [11].

На сегодняшний день клеточные автоматы нашли широкое распространение во всех сферах человеческой деятельности: математика, физика, биология [12], химия, механика, криптография [13], гидродинамика, газодинамика [14, 15] и т.д. Основным направлением применения клеточных автоматов является моделирование динамических процессов, например, движение толпы [16] или распространение тепловых потоков [17]. Некоторые ученые считают, что клеточные автоматы – это принцип, по которому сама природа действует в повседневной жизни. Наиболее подробно об этом описано в книге Стивена Вольфрама (Stephen Wolfram) «A New Kind of Science» [2].

Принцип клеточных автоматов основан на локальных взаимодействиях. Все элементы в любой рассматриваемой системе действуют по одинаковым принципам. В совокупности результаты работы простых правил, на которых основаны взаимодействия, позволяют получать сложное поведение. Джон

фон Нейман дает следующее определение клеточным автоматам: «Клеточные автоматы являются дискретными динамическими системами, поведение которых полностью определяется в терминах локальных зависимостей. В значительной степени также обстоит дело для большого класса непрерывных динамических систем, определенных уравнениями в частных производных. В этом смысле клеточные автоматы в информатике являются аналогом физического понятия «поля»… клеточный автомат может мыслиться как стилизованный мир. Пространство представлено равномерной сеткой, каждая ячейка или клетка которой содержит несколько битов данных; время идет вперед дискретными шагами, а законы мира выражаются единственным набором правил, скажем, небольшой справочной таблицей, по которой любая клетка на каждом шаге вычисляет своё новое состояние по состояниям её близких соседей. Таким образом, законы системы являются локальными и повсюду одинаковыми. «Локальный» означает, что для того, чтобы узнать, что произойдет здесь мгновение спустя, достаточно посмотреть на состояние ближайшего окружения: никакое дальнодействие не допускается. «Одинаковость» означает, что законы везде одни и те же: я могу отличить одно место от другого только по форме ландшафта, а не по какой-то разнице в законах» [1].

Формально клеточный автомат можно определить как набор $\{G, Z, N, f\}$, где G – метрика поля, на котором действует клеточный автомат;

Z – множество состояний каждой клетки;

N – окрестность клетки, которая влияет на состояние данной клетки;

f – правила клеточного автомата, которые в математическом виде могут быть записано $Z \, x \, Z^{|N|} \rightarrow Z$.

Свойствами клеточного автомата являются: локальность правил, однородность системы, конечность множества состояний клетки, одновременность изменений для всех клеток. Более подробно и формально

основные свойства классического клеточного автомата приведены в работе [18].

Клеточные автоматы могут работать в нескольких измерениях: он может обрабатывать вектор состояний клеток, плоскость или трехмерную модель. Во всех случаях для визуализации работы автомата часто добавляют дополнительное измерение – время, которое позволяет оценить изменения состояний поля в динамике.

Наиболее известным приложением клеточных автоматов можно считать игру «Жизнь» Джона Конвея (John Conway) [10, 19].

Существует несколько разновидностей клеточных автоматов. Ниже перечислены их основные типы из книги Стивена Вольфрама «A New Kind of Science» [2].

2.1.1. Стандартные клеточные автоматы

Стандартные клеточные автоматы на основе классического определения. Клетки поля такого автомата изменяют состояние в зависимости от состояний соседних клеток, и, таким образом, поле изменяет свое состояние. Примеры стандартных одномерных клеточных автоматов представлены на рис. 1.

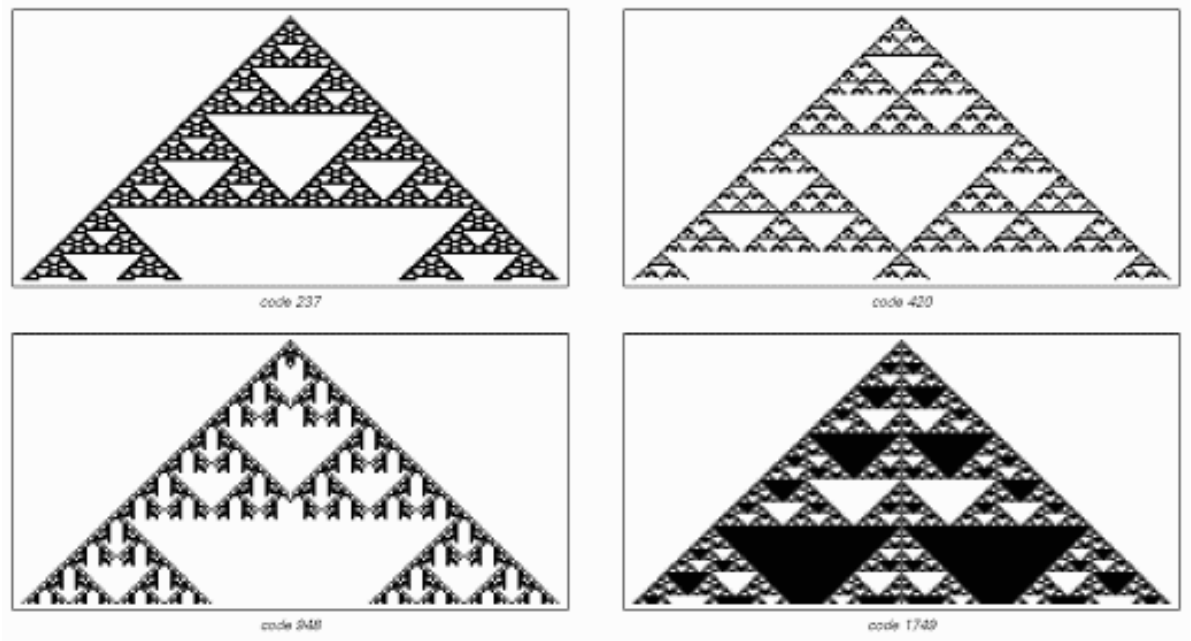


Рис. 1. Клеточные структуры, созданные с помощью стандартных клеточных автоматов

2.1.2. Мобильные автоматы

Мобильные клеточные автоматы подобно стандартным клеточным автоматам имеют состояния клеток, которые изменяются с течением времени, но в отличие от них дополнительно определена одна активная клетка. Только активная клетка изменяет состояние в каждый момент времени. Пример клеточной структуры, порожденной мобильным клеточным автоматом, изображен на рис. 2.

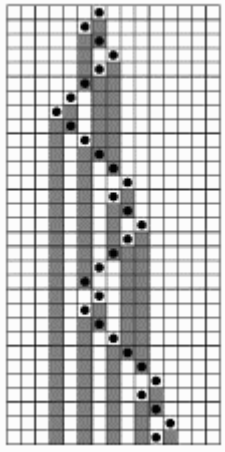


Рис. 2. Клеточная структура, созданная с помощью мобильного клеточного автомата

2.1.3. Тьюринговые машины

Тьюринговые машины наподобие мобильного клеточного автомата имеет только одну активную клетку. Дополнительное условие – определение нескольких состояний у активной клетки, как показано на рис. 3.

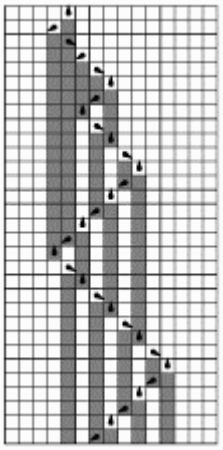


Рис. 3. Клеточная структура, созданная с помощью клеточного автомата – тьюринговой машины

2.1.4. Системы подстановок

Система подстановок – это клеточный автомат, который на каждом шаге заменяет клетку блоком из других клеток. Пример автомата показан на рис. 4.

Рис. 4. Клеточная структура, созданная с помощью системы подстановок

2.1.5. Последовательные подстановочные системы

Последовательные подстановочные системы сканируют последовательность точек на соответствие условиям каждого правила, при удовлетворении правила происходит подстановка и сканирование начинается заново, как изображено на рис. 5.

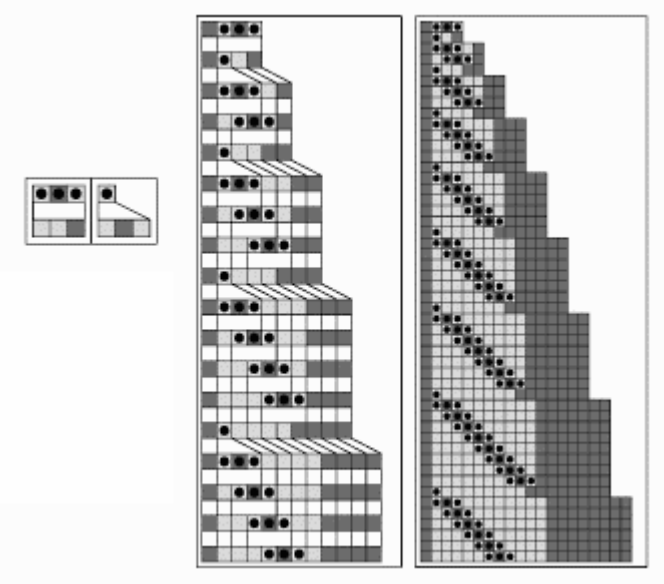


Рис. 5. Клеточная структура, созданная с помощью последовательной системы подстановок

2.1.6. Системы тэгов

Правила, которые удаляют элементы из начала цепочки клеток и добавляют другие клетки в конец цепочек, называются системами тэгов. Пример представлен на рис. 6.

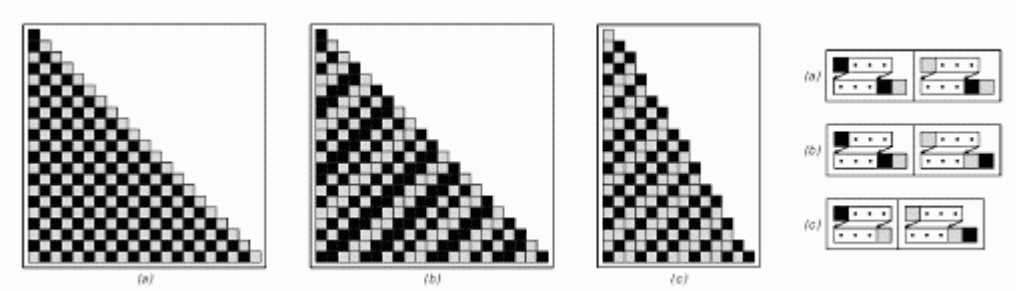


Рис. 6. Клеточная структура, созданная с помощью системы тэгов

2.1.7. Цикличные системы тэгов

Цикличность системы тэгов может достигаться за счет поочередного использования разных систем правил. В результате характер поведения клеточного автомата становится специфическим, что видно на рис. 7.

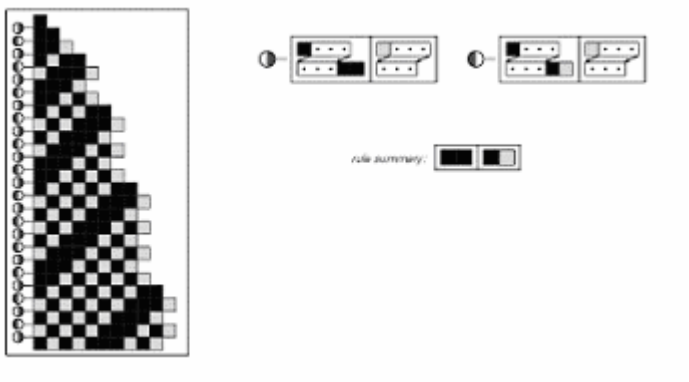


Рис. 7. Структура, созданная с помощью цикличной системы тэгов

2.1.8. Регистровые машины

Регистровые машины выполняют заданную программу. При успешном выполнении элемента программы происходит заданный переход, иначе переход на следующий элемент. Пример приведен на рис. 8.

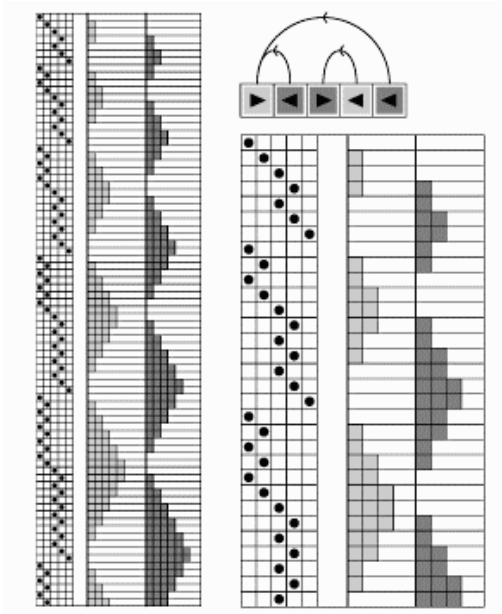


Рис. 8. Структура, созданная с помощью регистровой машины

2.1.9. Символьные системы

В символьной системе на каждом шаге каждый регион, заключенный в скобки, трансформируется по заданным правилам. В результате этого формируется последовательность. Пример показан на рис. 9.

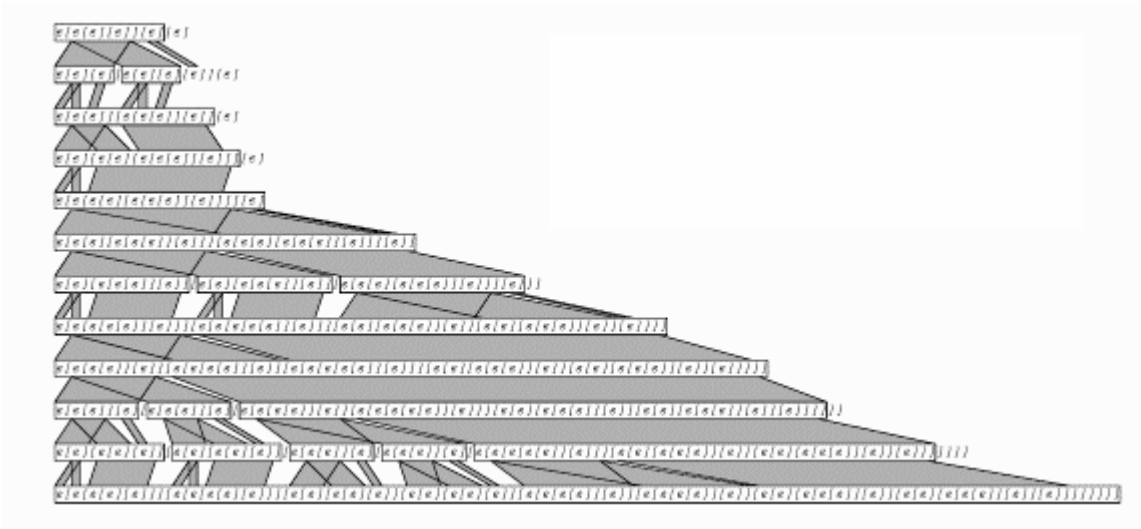


Рис. 9. Структура, созданная с помощью символьной системы

В данной работе (разд. 3.2) используется несколько типов клеточных автоматов. Особое место занимают мобильные клеточные автоматы, работающие на двумерной системе, а также клеточные системы с метками, которые введены в данной работе.

2.2. Клеточный автомат с метками

Понятие клеточного автомата с метками введено автором данной работы. Причиной его создания являлась необходимость достижения заданных характеристик клеточных автоматов при разработке алгоритмов процесса распознавания текста. Как будет показано далее, это понятие позволяет эффективно реализовать алгоритмы выделения признаков символов.

В процессе работы над дисертацией – анализа литературы – обнаружено понятие памяти в теории тьюринговых машин [20]. Оно определяет, что головка тьюринговой машины может хранить некоторое конечное число

данных, которые используются правилами автомата машины для изменения ленты машины или передвижения ее головки. Данное понятие при введении некоторых ограничений и перехода на теорию клеточных автоматов может быть также сведено к клеточным автоматам с метками.

Принцип клеточной системы с метками заключается в ассоциации каждой клетки поля с одним или несколькими метками.

Примером клеточного автомата с метками может служить набор: поле, на котором определены только клетки черного и белого цветов, метка, которая может присутствовать или отсутствовать в клетках поля, и правила, учитывающие наличие меток у соседних клеток. В случае определения одной метки для клеточного автомата, количество состояний каждой клетки увеличивается в два раза относительно случая без меток. В приведенном примере количество состояний каждой клетки равно четырем. Правило клеточного автомата в такой системе будут выглядеть так, как представлено на рис. 10.

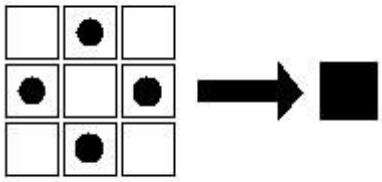


Рис. 10. Пример правила клеточного автомата с метками

В данном примере используется правило, основанное на восьми соседях. Без меток количество вариантов подобных правил (в случае определения только двух цветов клеток) будет $2^{10} = 1024$ (девять клеток условий и одна - результата). Добавляя метку, получится $(2 \cdot 2)^{10} = 1\ 024^2 = 1\,048\,576$ вариантов правил. Удобнее, в данном случае, разделять двоеточием нумерацию цветов и меток. Тогда для правила, представленного на рисунке

10, номером может служить 1:340 (или в двоичном виде 0000000001:0101010100, где каждая цифра определяет состояние клетки, клетки нумеруются слева направо сверху вниз), где первое число определяет правило преобразования цвета, а второе число – правило изменения метки.

Правило для поля, на котором определено более одной метки, нумеруется аналогичным образом: для правила с двумя метками – x : y : z, где первое число определяет правило преобразования цвета, второе число – преобразование одной метки, третье число – второй метки.

Клеточный автомат с метками позволяет получать сложные результаты без изменения структуры исспользуемой системы клеток. Например, для заданного изображения, данный автомат позволяет определить некоторые свойства данного изображения и выделить его характерные признаки без изменения цвета.

Формальное описание

Клеточный автомат с метками – это набор $\{G, M, Z, N, f\}$. Описание каждого элемента представлено ниже.

1. G – конечное дискретное метрическое множество, гарантирующее конечность расстояний между клетками.

2. M – конечное множество меток, определенное для каждой клетки.

3. Z – конечный набор состояний клеток.

4. N – конечное множество, определяющее окрестность клетки таким образом, что каждый элемент множества позволяет определить соседа для каждой клетки. |N| – количество соседних клеток, которые влияют на состояние данной.

5. f – правила клеточного автомата, соответствующие математической функции переходов: $Z \times C \times Z^{|N|} \times C^{|N|} \to Z \times C$, где $C \subset M$.

Ниже показано, что клеточный автомат с метками удовлетворяет требованиям классического клеточного автомата.

По определению клеточный автомат задается четырьмя элементами {G*, Z*, N*, f*} [18]. Элементы G и N клеточного автомата с метками соответствуют классическим элементам G* и N*.

Введем $Z^{\wedge} = Z\ x\ C$ таким образом, что $|Z^{\wedge}| = |Z|\ x\ |C|$. Множество $Z^{\wedge}$ будет соответствовать Z* классического определения клеточного автомата, так как оно является конечным и определяет все варианты состояний клетки автомата.

Аналогично, $f^{\wedge} = Z^{\wedge}\ x\ Z^{\wedge\ |N|} \rightarrow Z^{\wedge}$ будет соответствовать f*, так как учитываются все состояния на первом и втором временном слое.

Таким образом, получен набор {G, $Z^{\wedge}$, N, $f^{\wedge}$} клеточного автомата, в котором выполняются все свойства: локальность правил (обеспеченное N), однородность системы на основе метрики G, конечность множества состояний клетки $Z^{\wedge}$, одновременность изменений для всех клеток, гарантируемое набором правил $f^{\wedge}$.

Пример клеточного автомата с метками

На основе моделирующей программы, которая будет описана в главе 4, разработан пример клеточного автомата с метками «шахматная доска».

Задача автомата «шахматная доска» заключается в отрисовке чередующихся черно-белых точек на белом изображении.

Для выполнения задачи клеточному автомату с метками «шахматная доска» достаточно восьми правил: 0:257, 0:65, 0:321, 1:52, 513:52, 256:52, 64:52, 320:52 (нерассмотренные состояния девяти исходных клеток не изменяют состояние результирующей клетки). Эти правила изображены на рис. 11.

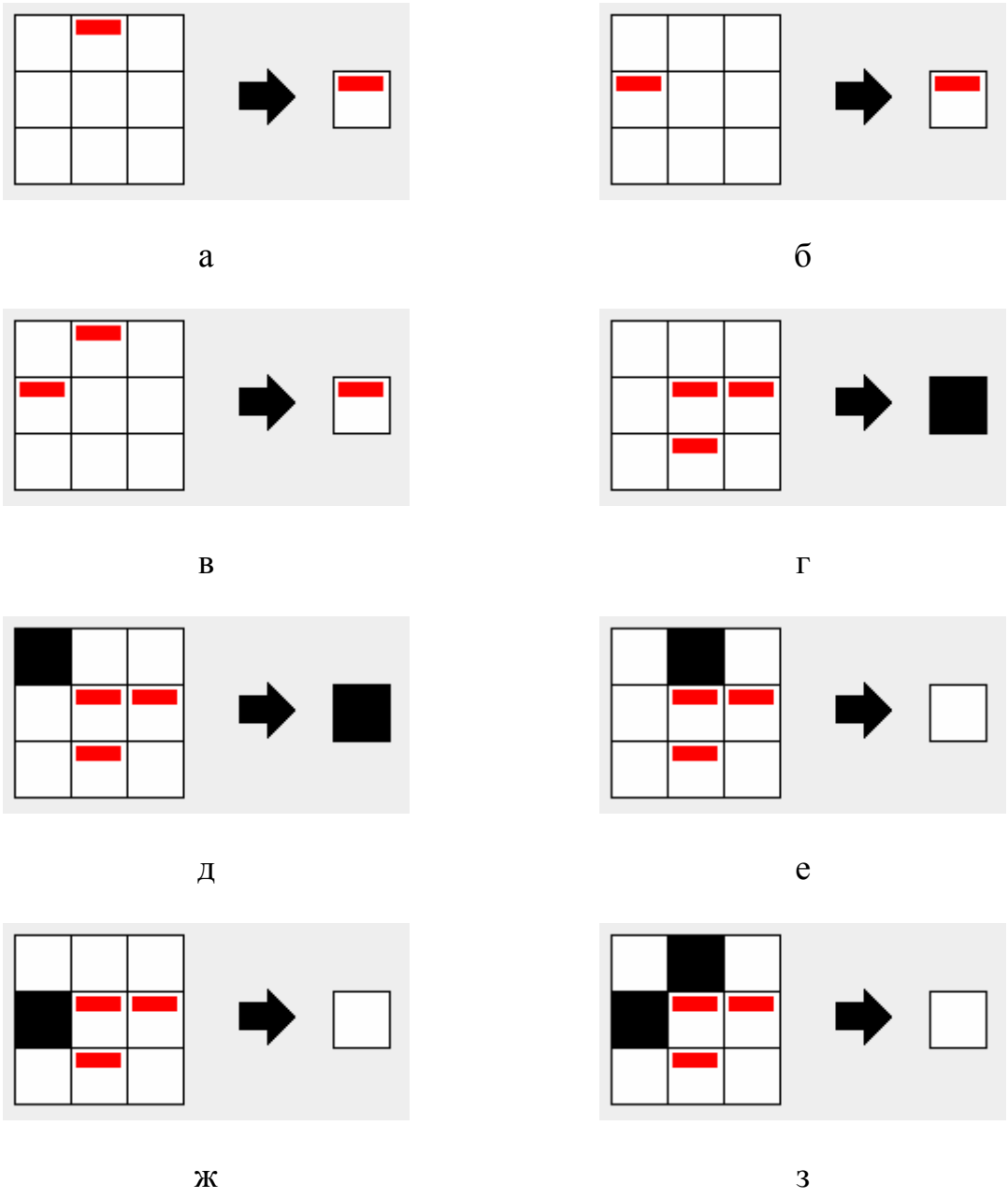
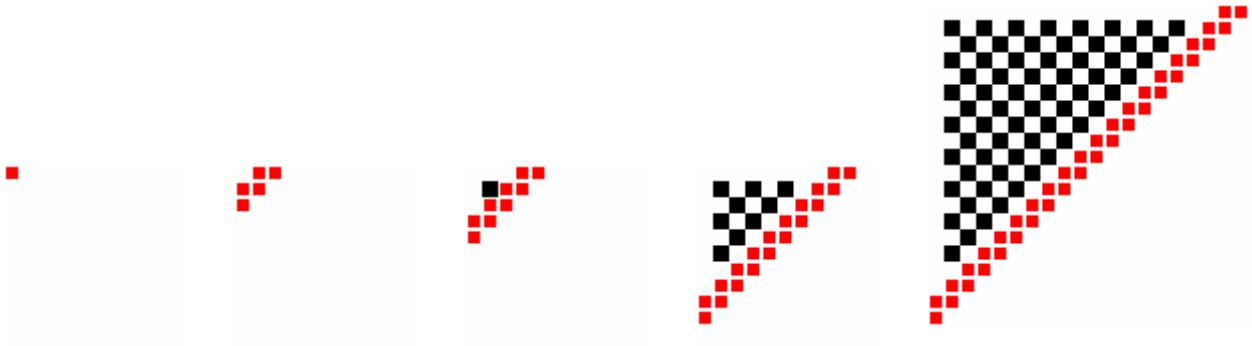


Рис. 11. Правила клеточного автомата «шахматная доска», правила: а – 0:257, б – 0:65, в – 0:321, г – 1:52, д – 513:52, е - 256:52, ж – 64:52, з – 320:52

Для начала работы автомата необходимо поставить метку на стартовой точке поля. Результат работы автомата на шагах 1, 3, 5, 10, 20 изображены на рис. 12.



а б в г д

Рис. 12. Результат работы клеточного автомата с метками «шахматная доска» на шагах: а – первом, б – третьем, в – пятом, г – десятом, д – двадцатом

Данный пример иллюстрирует применение клеточных автоматов с метками, но не затрагивает практическую полезность, которую можно обеспечить с их использованием: возможность определения характеристик, не изменяя цвета обрабатываемого изображения. Пример использования данного свойства будет приведен в разд. 3.2.

2.3. Последовательность клеточных автоматов

Клеточный автомат – это мощный инструмент по выполнению задач моделирования. Четко определив все его составляющие (метрику поля, состояния клеток, количество влияющих на клетку соседей и правила работы автомата), можно решить огромное число задач, многие из которых имеют не тривиальные решения [2].

К сожалению, не все задачи, решение которых возможно на основе клеточных автоматов, могут быть решены с помощью их небольшого числа. Более того, для решения части задач может потребоваться последовательность из клеточных автоматов, каждый из которых решает специализированную задачу.

Последовательность клеточных автоматов – это совокупность клеточных автоматов, каждый из которых основан на специфических правилах. Клеточные автоматы в последовательности запускаскаются в заданной последовательности и каждый очередной клеточный автомат начинает работу на поле, который был сформирован предыдущим клеточным автоматом в последовательности.

Последовательность автоматов не обязательно будет усложнять логику работы системы. В разд. 3.2 будет показано, что использование последовательности клеточных автоматов помогает обходиться небольшим числом правил для каждого клеточного автомата при решении нетривиальной задачи.

В случае использования клеточных автоматов с метками в последовательности клеточных автоматов необходимо заранее определить одинаковый набор M возможных меток клеток поля каждого автомата. Это поможет поддерживать последовательность в работоспособном состоянии.

ГЛАВА 3. КЛЕТОЧНЫЕ АВТОМАТЫ В СИСТЕМЕ РАСПОЗНАВАНИЯ ТЕКСТА

3.1. Клеточные автоматы в процессе распознавания

Способность "распознавать" считается основным свойством человека, а также других живых организмов. Образ представляет собой описание объекта. В каждое мгновение бодрствования совершаются акты распознавания.

Распознавание человеком конкретных образов можно рассматривать как психофизиологическую задачу, связанную с процессом взаимодействия индивида с определенным физическим раздражителем. Когда индивид воспринимает образ, он реализует процесс индуктивного вывода и устанавливает ассоциативную связь между своим восприятием и определенными обобщенными понятиями или "ориентирами", установленными им на основании прошлого опыта. В сущности, распознавание человеком образов можно свести к вопросу оценки относительных шансов на то, что исходные данные соответствуют тому или иному из известных множеств статистических совокупностей, определяющихся прошлым опытом человека и предоставляющих ориентиры и априорную информацию для распознавания [21]. Таким образом, задача распознавания образов можно рассматривать как задачу установления различий между исходными данными, причем не посредством отождествления с отдельными образами, но с их совокупностями. Это отождествление осуществляется при помощи поиска признаков (инвариантных свойств) на множестве объектов, образующих определенную совокупность.

Поиск признаков является одним из самых важных этапов в процессе распознавания образов и, в частности, символов. Описание некоторых

решений задачи по выделению признаков символов на основе клеточных автоматов приводится в разд. 3.2.

До процесса выделения признаков символов необходимо, как указано в разд.1.1, решить несколько задач: необходимо изображение текста обработать от шума, привести его в состояние, которое позволяет выполнить условия алгоритмов распознавания, и выделить из него отдельные изображения символов.

3.1.1. Предварительная обработка изображения

Обработка изображения – это задача по изменению характеристик изображения для того, чтобы алгоритмы, участвующие в распознавании текста, работали более качественно с меньшим числом ошибок. Кроме того, для увеличения производительности работы клеточных автоматов, задействованных в распознавании, очень критичным элементом является количество состояний клеток (цветов точек) изображения.

В данной работе клеточные автоматы, задействованные в распознавании символов, функционируют на основе двух состояний клеток, соответствующих черному и белому цвету пикселей изображения.

В процессе перевода изображения в черно-белое состояние составляющие символов точки следует выделить среди фона. Для данной задачи может быть использован клеточный автомат, в котором каждая клетка соответствует точке изображения, а локальный радиус для клетки равен нулю. Автомат реализует три правила: переводит цвет каждой точки изображения в оттенок серого; закрашивает клетку черным, если она темнее определенного предельного цвета; закрашивает клетку белым цветов, если она светлее установленной границы. На рис. 13 представлены правила описанного клеточного автомата, созданные на основе моделирующей программы, описанной в главе 4.

а

б

в

Рис. 13. Правила автомата перевода изображения текста в черно-белые цвета: а – перевод цвета точки в оттенок серого, б – закрашивание клетки черным, если ее оттенок больше предельного цвета, в - закрашивание клетки белым, если ее оттенок меньше предельного цвета

3.1.2. Деление текста на символы

Одна из нерешенных на сегодняшний день задач для систем распознавания текста – его деление на символы [4].

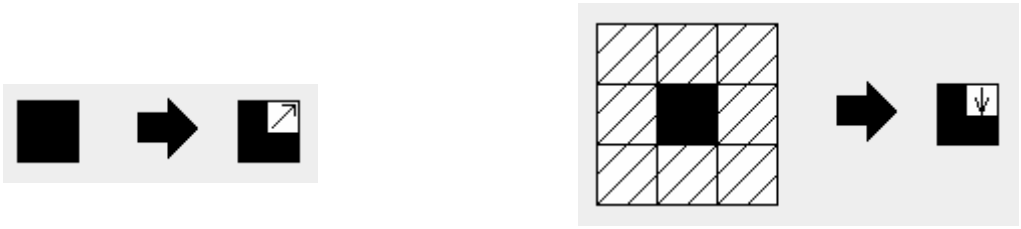
Наиболее часто эта задача решается в подобных системах в комплексе: производится предварительно разбиение изображения текста на отдельные изображения символов и после этого определяется зависимость между разными изображениями с оценкой расстояния между ними. На этапе распознавания результаты разбиения могут уточняться с целью дополнительного разбиения или объединения нескольких полученных изображений. Уточнения могут строиться на основе контекста: если часть символов хорошо распознаны, они могут указать на нераспознанный, либо факт уточнения может основываться на плохом распознании получившегося изображения символа.

Клеточные автоматы могут помочь в предварительном разбиении изображения текста на изображения символов. В этом случае необходимо определить два клеточных автомата с метками, правила которых описаны ниже.

1. Первый автомат ставит метку на каждую черную точку изображения в виде последовательно генерируемых целых чисел.

2. Второй автомат для каждой черной точки просматривает локальную окрестность единичного радиуса и саму себя, выставляет у себя метку с минимальным числом из этой окрестности. При этом старая метка удаляется.

Схемы автоматов, созданных на основе моделирующей программы, представлены на рис. 14.



а б

Рис. 14. Клеточные автоматы с метками, выделяющие из изображения текста символы: а – автомат генерации меток – чисел, б – автомат по поиску метки с минимальным числом

После завершения работы автоматов в изображении текста будут выделены разными метками разные символы, что позволит вычленять изображения отдельных символов.

3.1.3. Выделение признаков символов

Выше было отмечено, что человек воспринимает образы на основе ассоциации признаков оцениваемых объектов со знакомыми объектами. Каждый символ текста (русского, английского или другого) имеет свои уникальные признаки. Данные признаки уникальным образом отличают символы друг от друга.

Символы текста имеют большое число признаков: положение и наклон линий, дуг, наличие петель, вертикальных – горизонтальных линий, выступы и их наклон, пересечения. Основными признаками можно считать выступы, петли и пересечения, а также их взаимное расположение. На рис. 15 представлено положение элементов этих трех видов в символах русского алфавита. На рисунке не представлены символы, являющиеся объединением нескольких несвязанных элементов. Это символы «Ё», «Й» и «Ы». В настоящей роботе данные символы не расматриваются, для их распознавания в алгоритмах должны присутствовать дополнительные специальные блоки.

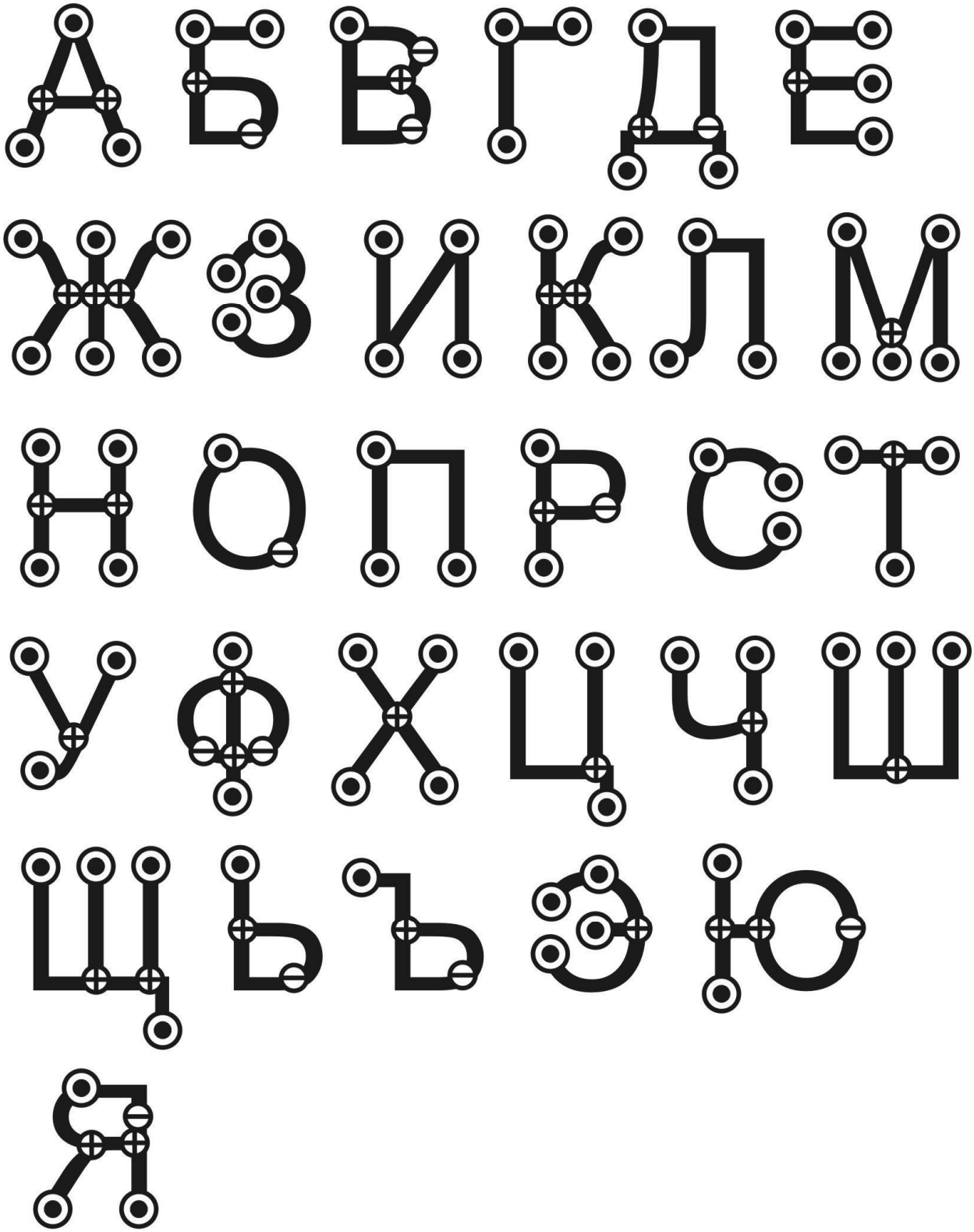


Рис. 15. Положение петель, концов и пересечений в символах русского алфавита

Решение задачи выделения признаков символов на основе клеточных автоматов решается в разд. 3.2.

3.1.4. Классификация и обучение

На предыдущем шаге процесса распознавания текста выделяются признаки символов. После этого предполагается процесс классификации, который на основании полученных признаков определит название символа.

Классификация, как было отмечено в разд. 1.2, наиболее часто основывается на нейронных сетях. Кроме того, для этой цели применимы статистические методы, которые на основе накопленной информации о признаках могут определить символ.

Система распознавания предполагает наличие блока обучения. Обучение системы напрямую связано с классификацией, оно позволяет изменять и поправлять коэффициенты на основе ассоциации результата классификации с названием символа. Для нейронных сетей и статистических методов обучение производится различными способами [5, 22].

Клеточные автоматы также могут участвовать в процессе классификации признаков. Идеей классификации может служить создание характерного клеточного автомата для каждого признака и его коррекция с учетом определенных признаков в процессе обучения. Данная концепция требует изучения и дальнейших исследований.

3.2. Клеточные автоматы в процессе выделения свойств текста

В работе основное внимание уделено изучению характеристик клеточных автоматов в процессе распознавания текста. Разбиение процесса распознавания текста на этапы предполагает использование разных

клеточных автоматов для выполнения различных задач. Определение характеристик символов и выделение их признаков требует разработки комплекса правил, на основе которых это становится возможным.

В разд. 3.1 описано, что основными элементами символов являются петли, пересечения, положение концов. По этим элементам возможна идентификация изображения, соотнесение с конкретным символом. Возможно, что человек подсознательно пользуется именно этими признаками в процессе чтения текста.

Существует множество стратегий выделения описанных признаков на основе клеточных автоматов. Ниже описаны две таких стратегии, которые используют клеточные автоматы с метками.

3.2.1. Первая стратегия выделения концов и петель символа

Первая стратегия состоит в том, что от верхнего края символа вдоль точек, составляющих данный символ, пускается «волна». Эта «волна» может разделяться на составляющие, повторяя контур изображения символа. В определенный момент составляющие «волны» могут встретиться или затухнуть на конце символа. На рис. 16 показано направление движения «волны» вдоль изображения символа «А» и показаны места старта «волны», затухания на концах символа и в месте встречи ее двух составляющих.

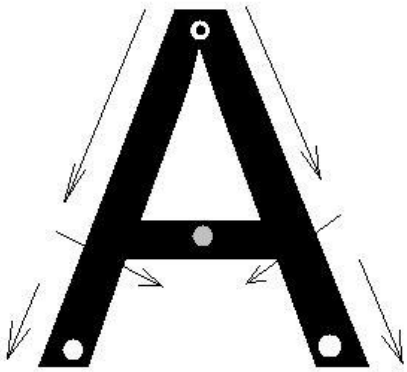


Рис. 16. Направление распространения «волны» вдоль символа «А» с отмечанием позиций концов символов и петель

В понятие «волны» в данном алгоритме вкладывается несколько составляющих. «Фронт волны» – точки символа, которые передвигаются от одного конца изображения символа к другому. «Шлейф волны» – это точки изображения, в которых в предыдущий момент времени находился «фронт волны». «Точки пройденного пути» – это точки изображения, где присутствовал «фронт волны», а затем и «шлейф волны», в этих точках процесс не возобновляется.

В начальный момент времени все точки символа не помечены метками. Первая точка волны помечается, и алгоритм начинает свою работу.

Алгоритм основан на идее, что в процессе прохождения волны в какой-то момент «фронт волны» угаснет, в то время как шлейф все еще будет присутствовать. Данное событие может случиться только на конце символа либо на месте встречи двух составляющих волны. Позиция шлейфа волны в этот момент запоминается.

Дополнительно, событие встречи двух составляющих волны регистрируется на основании факта, что шлейфы двух составляющих волны в момент встречи не связаны между собой. Таким образом, запоминается позиция петли символа.

Схема работы последовательности клеточных автоматов представлена на рис. 17. В овальных блоках указаны клеточные автоматы.

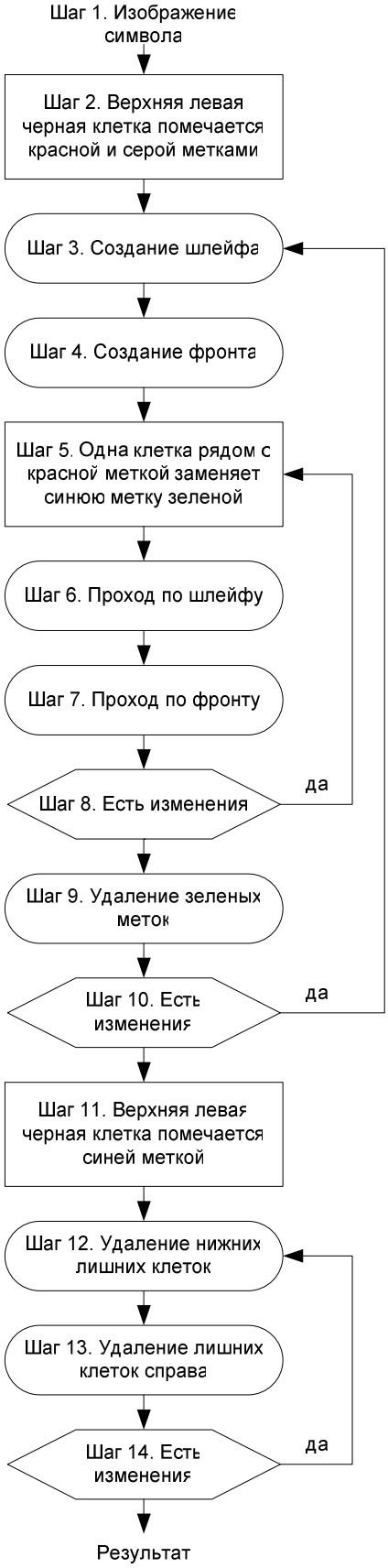


Рис. 17. Схема работы последовательности клеточных автоматов для первого алгоритма выделения признаков символов

Приведем описание алгоритма на основе последовательности клеточных автоматов с метками. Кроме клеточных автоматов в последовательности присутствуют дополнительные управляющие элементы. Состояние клетки определяется ее цветом. Кроме цвета, клетка может содержать одну или более меток, которые разделяются между собой цветами.

Шаг 1. На вход последовательности клеточных автоматов поступает изображение символа.

Шаг 2. Верхняя левая черная клетка изображения помечается красной и серой метками.

Шаг 3. Автомат «создание шлейфа»: красная метка заменяется синей меткой.

Шаг 4. Автомат «создание фронта»: черные клетки без серой метки рядом с клетками с серой меткой помечаются серой и красной метками.

Шаг 5. Одна клетка рядом с красной меткой заменяет синюю метку зеленой.

Шаг 6. Автомат «проход по шлейфу»: все клетки рядом с клетками с зелеными метками заменяют синие метки зелеными.

Шаг 7. Автомат «проход по фронту»: все клетки с красной, но без зеленой метки, находящиеся рядом с клетками с зеленой меткой, помечаются зеленой меткой; если клетка с красной и зеленой меткой находится рядом с клеткой с синей меткой, то данная клетка помечается оранжевой меткой.

Шаг 8. Если, начиная с шага 5, автоматы не изменили состояния ни одной клетки, то перейти на шаг 9, иначе перейти на шаг 5.

Шаг 9. Автомат «удаление зеленых меток»: все зеленые метки у клеток удаляются.

Шаг 10. Если, начиная с шага 3, автоматы не изменили состояния ни одной клетки, то перейти на шаг 11, иначе перейти на шаг 3.

Шаг 11. Верхняя левая черная клетка изображения помечается синей меткой.

Шаг 12. Автомат «удание нижних лишних меток»: если над клеткой с синей или оранжевой меткой находится клетка с такой же меткой, то в данной клетке эту метку удалить.

Шаг 13. Автомат «удаление лишних меток справа»: если слева от клетки с синей или оранжевой меткой находится клетка с такой же меткой, то в данной клетке эту метку удалить.

Шаг 14. Если, начиная с шага 12, автоматы не изменили состояния ни одной клетки, то завершить последовательность, иначе перейти на шаг 12.

На выходе алгоритма некоторые клетки изображения помечены метками. Синяя метка означает, что в данном месте символа находится конец символа, а оранжевая метка означает у символа петли.

В процессе работы алгоритма клетки с красными метками могут быть проинтерпретированы как фронт волны, синие клетки – как шлейф волны, а серые клетки – как клетки пройденного пути.

Ниже приведены формальные правила переходов клеточных автоматов с метками. Для описываемых клеточных автоматов определены пять различных меток: красного, синего, серого, зеленого и оранжевого цветов. Это означает, что состояние клетки состоит из шести составляющих: цвета и наличия/отсутствия пяти меток.

Номер правила обозначается шестью числами, которые означают состояние клетки и присутствие/отсутствие метки. Позиция каждой метки в номере определяется на основе приведенного выше списка.

Некоторые правила автоматов в схемах содержат вместо цвета символа штрихи – это означает набор правил со всевозможными комбинациями цвета/наличия меток клетки.

1. Автомат «создание шлейфа» имеет окрестность *N* радиуса ноль – на состояние клетки влияет только сама клетка. Автомат содержит только одно правило: 3 : 2 : 1 : 1 : 0 : 0. Схема правила указана на рис. 18.

Рис. 18. Схема правила клеточного автомата «создание шлейфа»

2. Автомат «создание фронта» имеет локальную окрестность с радиусов в одну клетку и содержит несколько правил. Схемы правил представлена на рис. 19.

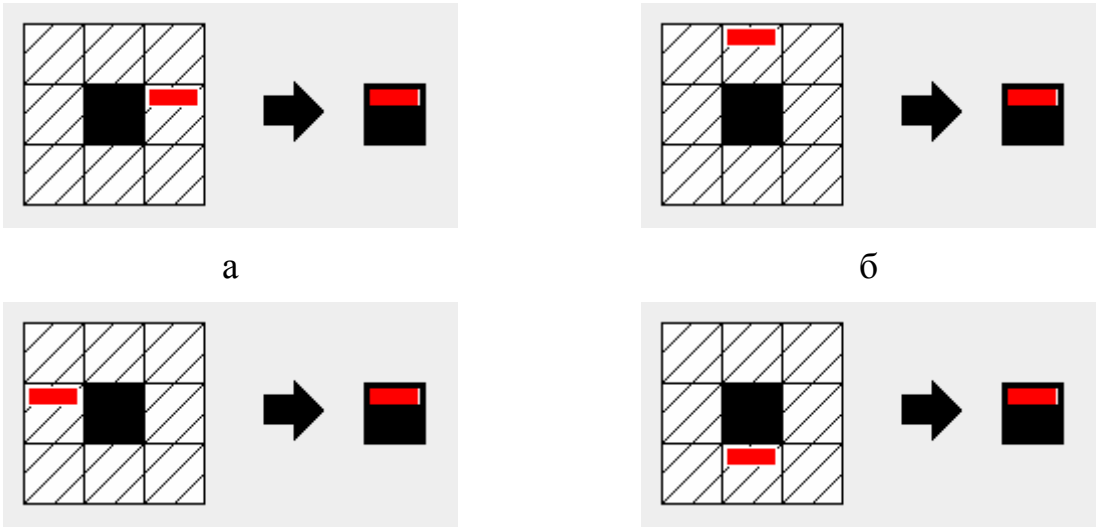


Рис. 19. Схемы правил клеточного автомата «создание фронта»

3. Автомата «проход по шлейфу» основан на окресности из соседей единичного радиуса и содержит восемь правил, схемы которых представлены на рис. 20.

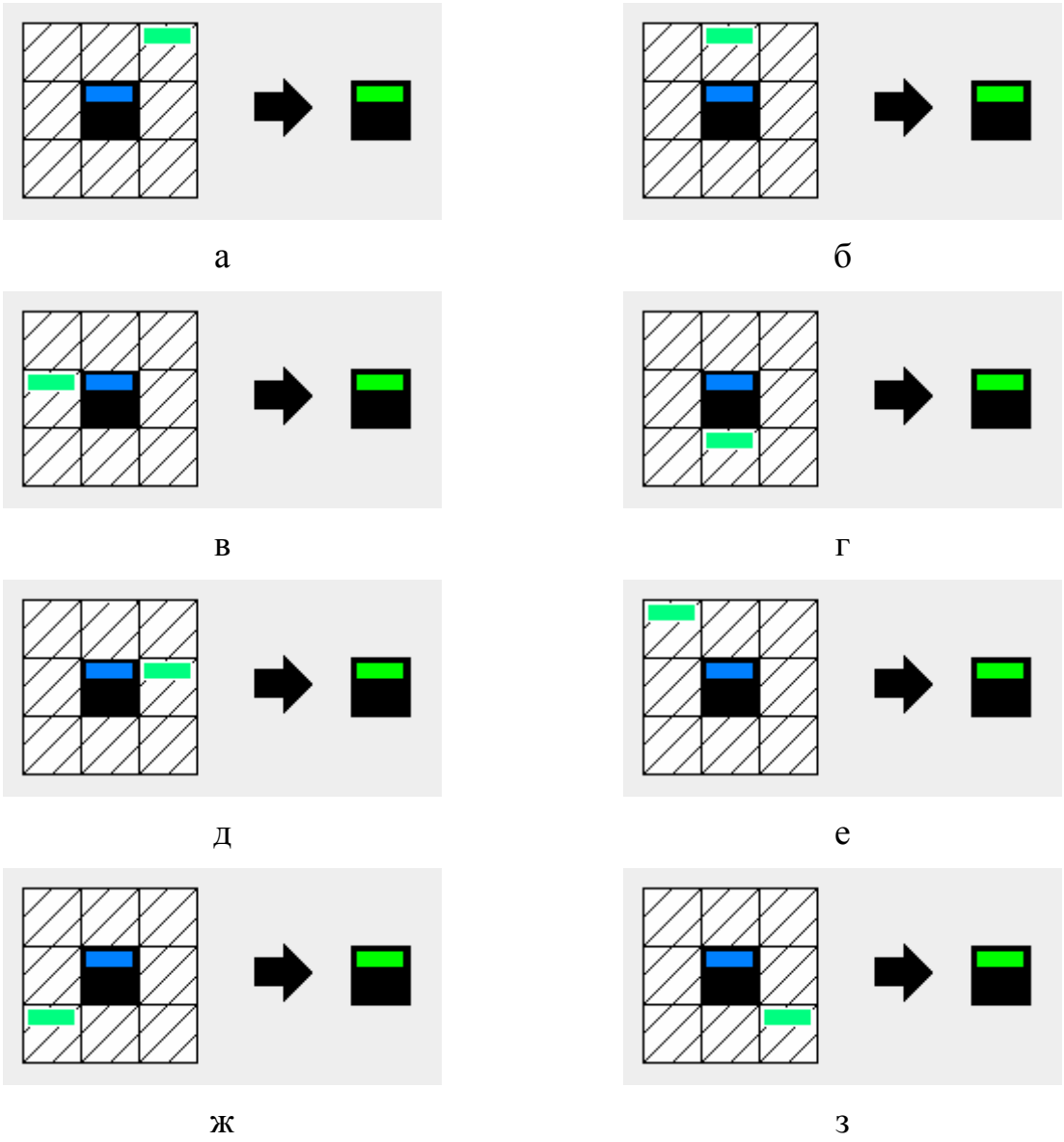


Рис. 20. Схемы правил клеточного автомата «проход по шлейфу»

4. Схемы автомата «проход по фронту» представлены на рис. 21, автомат содержит 12 правил.

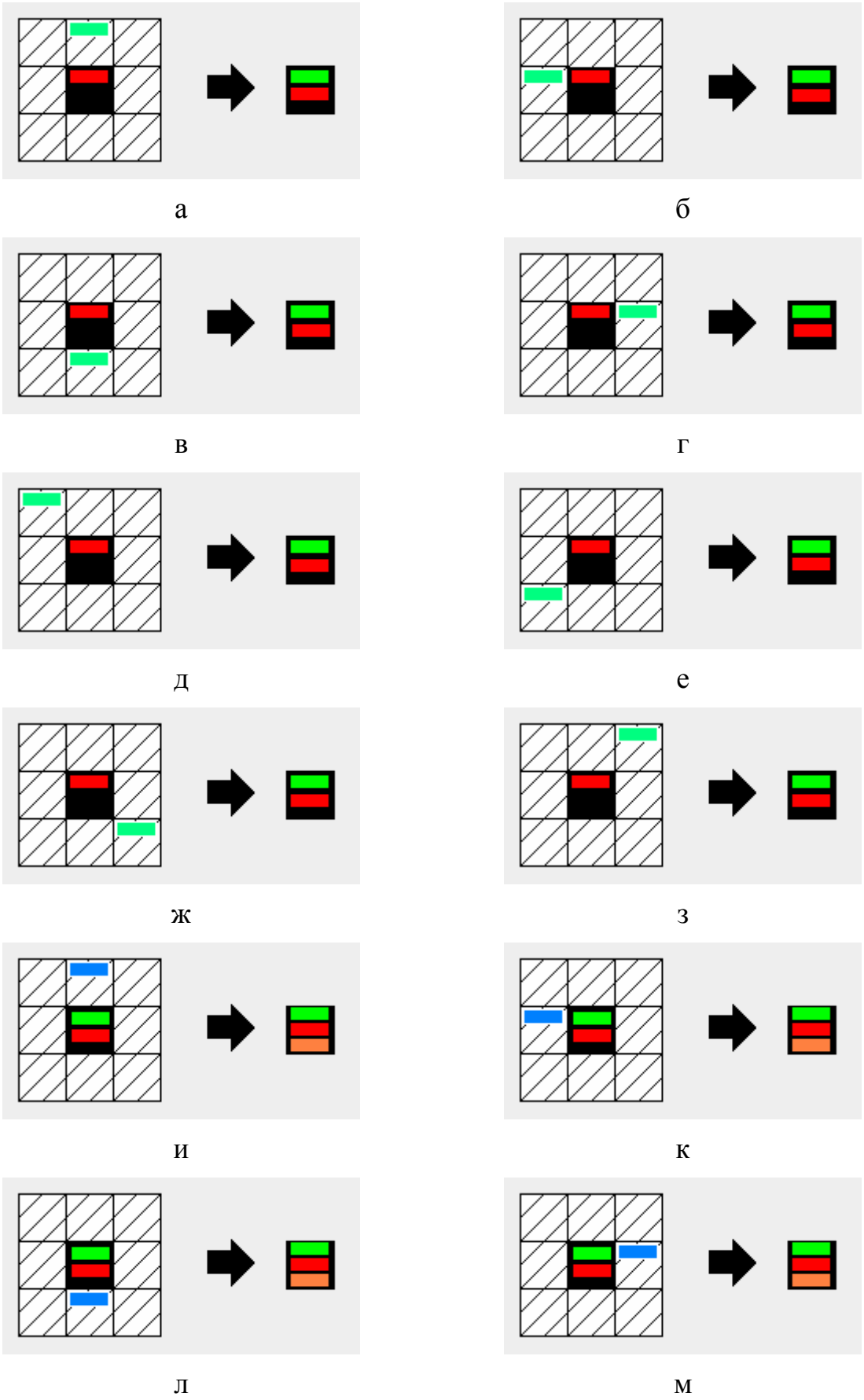


Рис. 21. Схемы правил клеточного автомата «проход по фронту»

5. Автомат «удаление зеленых меток» аналогично автомату «создание шлейфа» содержит одно правило и имеет радиус локальной окресности – ноль. Схема правила автомата показана на рис. 22.

Рис. 22. Схемы правил клеточного автомата «удаление зеленых меток»

6. Схемы правил клеточного автомата «удаление нижних лишних меток» показана на рис. 23.

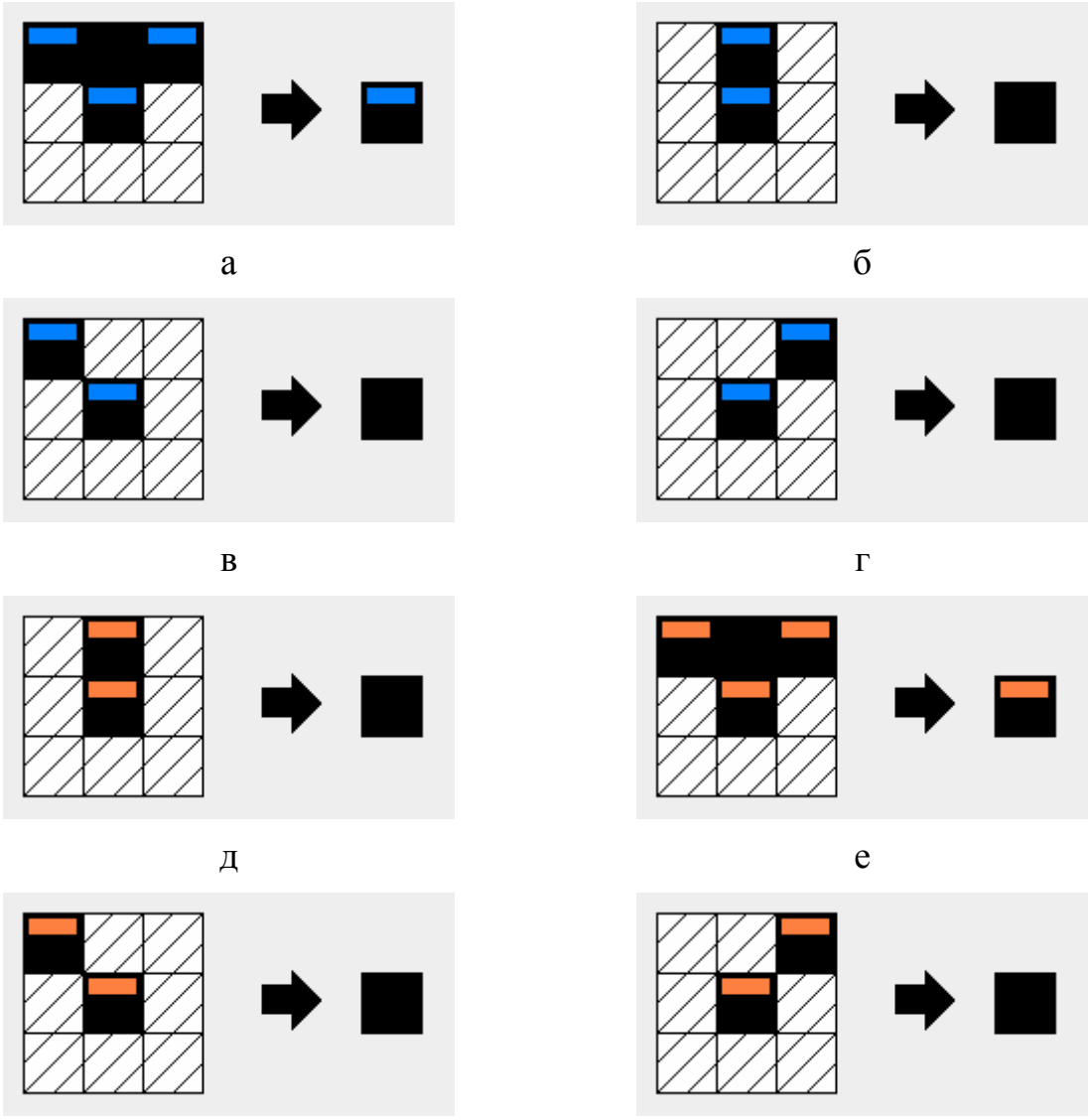


Рис. 23. Схемы правил автомата «удаление нижних лишних меток»

7. Схемы правил автомата «удаление лишних меток справа» показана на рис. 24.

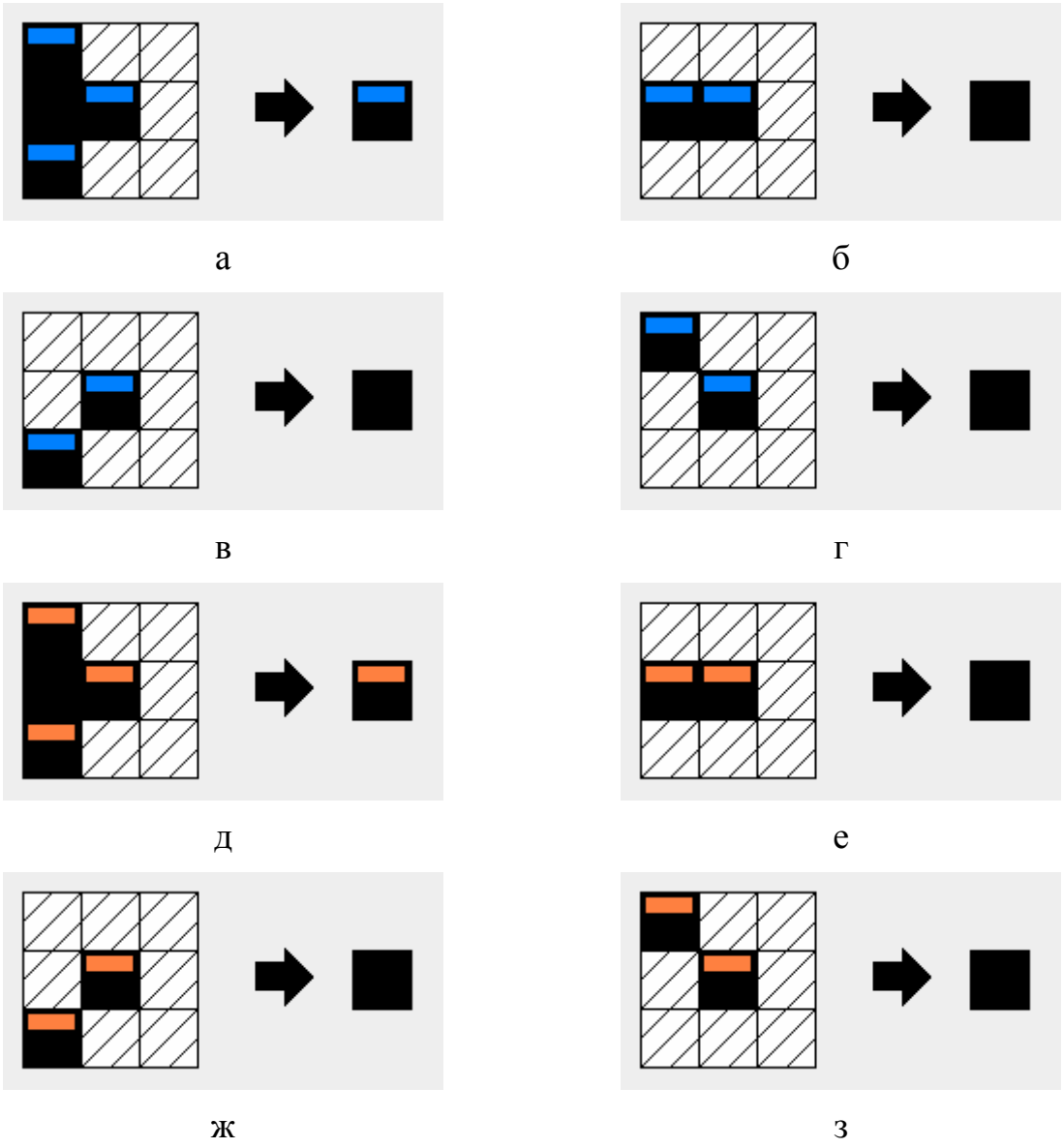


Рис. 24. Схемы правил клеточного автомата «удаление лишних меток справа»

3.2.2. Вторая стратегия выделения пересечений, концов и петель символа

Вторая стратегия представляет собой усовершенствованную первую и развивает ее. В процессе распространения «волны» в момент, когда определяется клетка конца символа или клетка встречи двух составляющих

«волны», генерируется ответная «волна» – «эхо» вдоль уже пройденных клеток. «Эхо» проходит путь в обратном направлении и отмечает клетки, в которых начальная волна была разделена на составляющие. Это должно позволить найти позиции пересечений отрезков, из которых состоят символы.

Вторая стратегия позволяет определить позиции пересечений линий в изображении символа. Распространение волны можно увидеть на рис. 25.

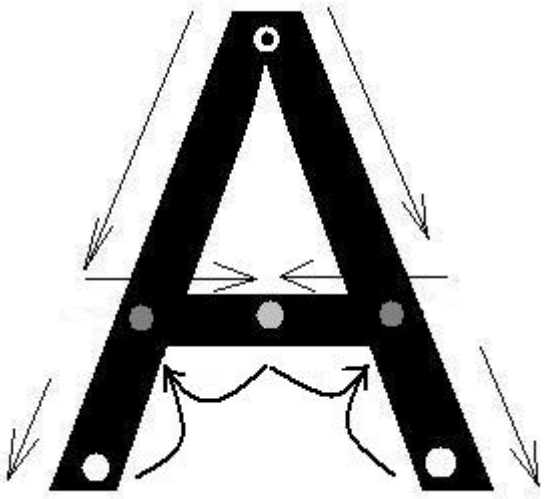


Рис. 25. Направление распространения «волны» вдоль символа «А» с отмечанием позиций концов символов и петель, а также распространение «эхо» с отмечанием позиции пересечений

Схема работы последовательности клеточных автоматов показана на рис. 26.

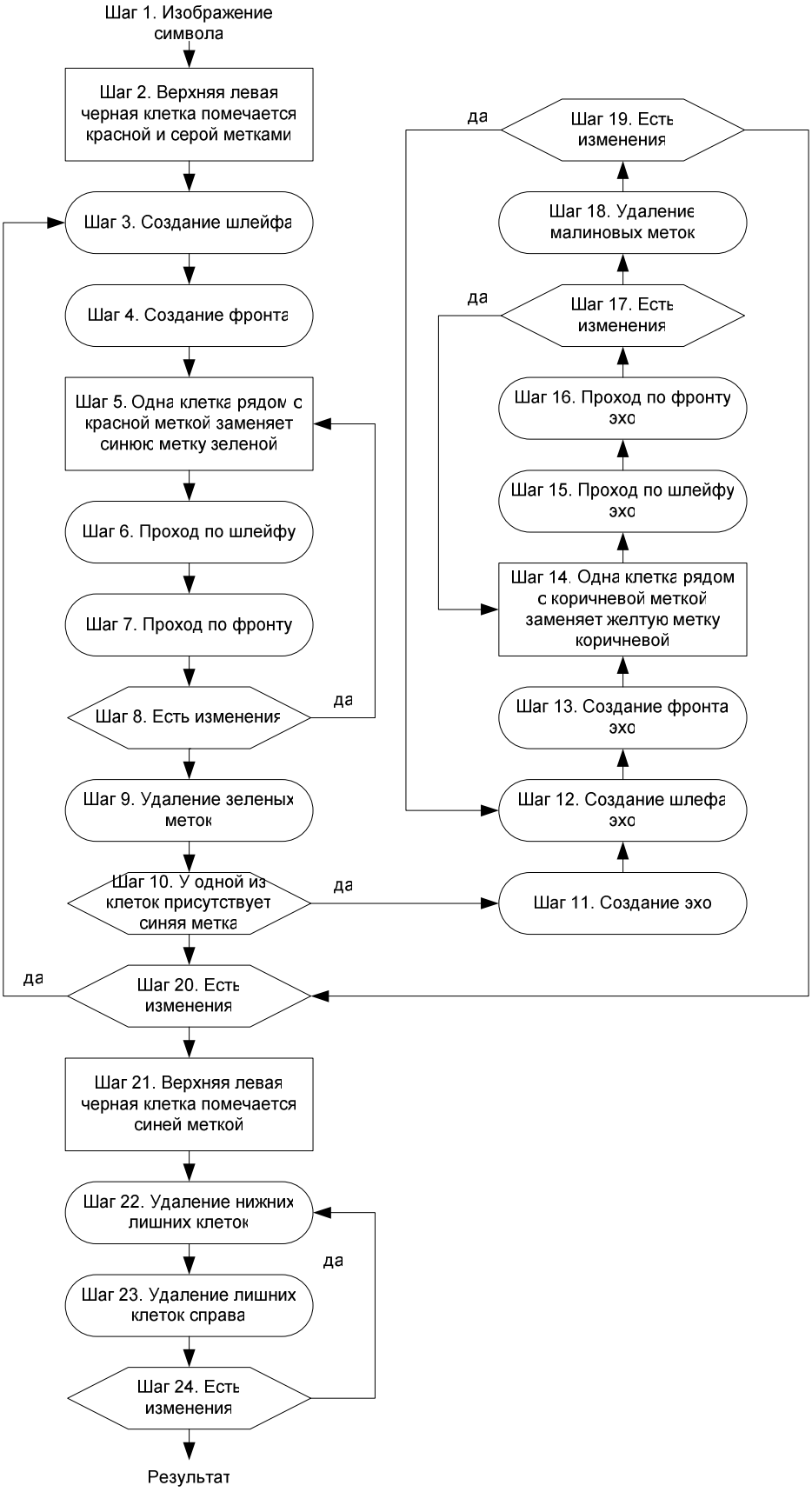


Рис. 26. Схема работы последовательности клеточных автоматов для второго алгоритма выделения признаков символов

Приведем неформальное описание второго алгоритма на основе последовательности клеточных автоматов с метками.

Шаг 1. На вход последовательности клеточных автоматов поступает изображение символа.

Шаг 2. Верхняя левая черная клетка изображения помечается красной и серой метками.

Шаг 3. Автомат «создание шлейфа»: красная метка заменяется синей меткой.

Шаг 4. Автомат «создание фронта»: черные клетки без серой метки рядом с клетками с серой меткой помечаются серой и красной метками.

Шаг 5. Одна клетка рядом с красной меткой изменяет синюю метку зеленой.

Шаг 6. Автомат «проход по шлейфу»: все клетки рядом с клетками с зелеными метками заменяют синие метки зелеными.

Шаг 7. Автомат «проход по фронту»: все клетки с красной, но не с зеленой меткой, находящиеся рядом с клетками с зеленой меткой, помечаются зеленой меткой; если клетка с красной и зеленой метками находится рядом с клеткой с синей меткой, то данная клетка помечается оранжевой меткой.

Шаг 8. Если, начиная с шага 5, автоматы не изменили состояние ни одной клетки, то перейти на шаг 9, иначе перейти на шаг 5.

Шаг 9. Автомат «удаление зеленых меток»: все зеленые метки у клеток убираются.

Шаг 10. Если у одной из клеток присутствует синяя метка, то перейти на шаг 11, иначе на шаг 20.

Шаг 11. Автомат «создание эха»: синюю метку клеток заменить коричневой и голубой, к красной метке символов добавить коричневую метку.

Шаг 12. Автомат «создание шлейфа эха»: коричневая метка заменяется желтой меткой.

Шаг 13. Автомат «создание фронта эха»: черные клетки с серой меткой, но без фиолетовой метки рядом с клетками с фиолетовой меткой помечаются фиолетовой и коричневой метками.

Шаг 14. Одна клетка рядом с коричневой меткой меняет желтую метку малиновой.

Шаг 15. Автомат «проход по шлейфу эха»: все клетки рядом с клетками с малиновыми метками заменяют желтые метки малиновыми.

Шаг 16. Автомат «проход по фронту эха»: все клетки с коричневой, но не с малиновой меткой, находящиеся рядом с клетками с малиновой меткой, помечаются малиновой меткой; если клетка с коричневой и малиновой метками находится рядом с клеткой с желтой меткой, то данная клетка помечается пурпурной меткой.

Шаг 17. Если, начиная с шага 14, автоматы не изменили состояния ни одной клетки, то перейти на шаг 18, иначе перейти на шаг 14.

Шаг 18. Автомат «удаление малиновых меток»: все малиновые метки у клеток убираются.

Шаг 19. Если, начиная с шага 12, автоматы не изменили состояния ни одной клетки, то перейти на шаг 20, иначе перейти на шаг 12.

Шаг 20. Если, начиная с шага 3, автоматы не изменили состояния ни одной клетки, то перейти на шаг 21, иначе перейти на шаг 3.

Шаг 21. Верхняя левая черная клетка изображения помечается синей меткой.

Шаг 22. Автомат «удание нижних лишних меток»: если над клеткой с синей или оранжевой меткой находится клетка с соответственно синей или оранжевой меткой, то в данной клетке эту метку удалить.

Шаг 23. Автомат «удание лишних меток справа»: если слева от клетки с синей или оранжевой меткой находится клетка с соответственно синей или оранжевой меткой, то в данной клетке эту метку удалить.

Шаг 24. Если, начиная с шага 22, автоматы не изменили состояния ни одной клетки, то завершить последовательность, иначе перейти на шаг 22.

Второй алгоритм может показаться, на первый взгляд, громоздким, но, на самом деле, он содержит в себе две составляющих, аналогичных первому алгоритму. Правила автоматов данного алгоритма аналогичны правилам автоматов первого алгоритма по выделению признаков символов.

Достоинством второго алгоритма можно считать выделение большего количества признаков символов, чем с помощью первого алгоритма. Однако, второй алгоритм выполняет больше шагов и поэтому работает дольше.

ГЛАВА 4. РЕАЛИЗАЦИЯ ПРАВИЛ КЛЕТОЧНЫХ АВТОМАТОВ

4.1. Архитектура моделирующей программы

Для проведения исследований по применению клеточных автоматов, клеточных автоматов с метками и последовательностей клеточных автоматов была разработана моделирующая программа на языке программирования *Java*. Архитектура программы позволила создавать, сохранять и загружать клеточные автоматы с метки и без них, а также последовательности клеточных автоматов.

Ниже описана структура основных частей программы, которые напрямую связаны с работой клеточных автоматов. Архитектура функциональной части, относящейся к работе программы в целом, приведена в приложении.

Каждая сущность в программе на основе принципов объектно-ориентированного проектирования представляется в виде класса с набором свойств и действий, выполняемых ею. Например, клеточный автомат, правило и клетка клеточного автомата являются классами.

4.1.1. Реализация клеточного автомата

Клеточный автомат можно определить как набор правил, причем у клеточного автомата должно быть определено хотя бы одно правило. Поэтому автомат может содержать правило по умолчанию и, возможно, остальные правила.

Каждое правило – это совокупность проверки состояния клетки и соседних клеток и определение нового состояния клетки. Если определен локальный радиус, на котором соседние клетки влияют на состояние данной, равный единице, то правило будет содержать девять клеток, которые

необходимо проверить на удовлетворение условиям правила, и результат действия правила – результирующая клетка.

Проверка клетки на удовлетворение условиям правила разделяется на проверку состояния клетки – его цвета и на проверку наличия или отсутствия определенных меток. Проверка цвета клетки может происходить несколькими способами: клетка должна иметь четко определенный цвет, цвет темнее данного, цвет светлее данного, цвет на отрезке и любой цвет. Аналогично проверка меток задается проверкой на наличие всех или одной из заданных меток, отсутствием всех или определенных меток.

Определение нового состояния клетки означает определение цвета и меток, которые будут определены для клетки как результат выполнения правила. Цвет клетки может быть оставлен прежним или четко заданным. Аналогично, метки могут быть оставлены прежними или добавлены новые. Дополнительно метки, найденные в клетке на этапе проверки в рамках данного правила могут быть удалены.

Диаграмма классов блока клеточного автомата представлена на рис. 27.

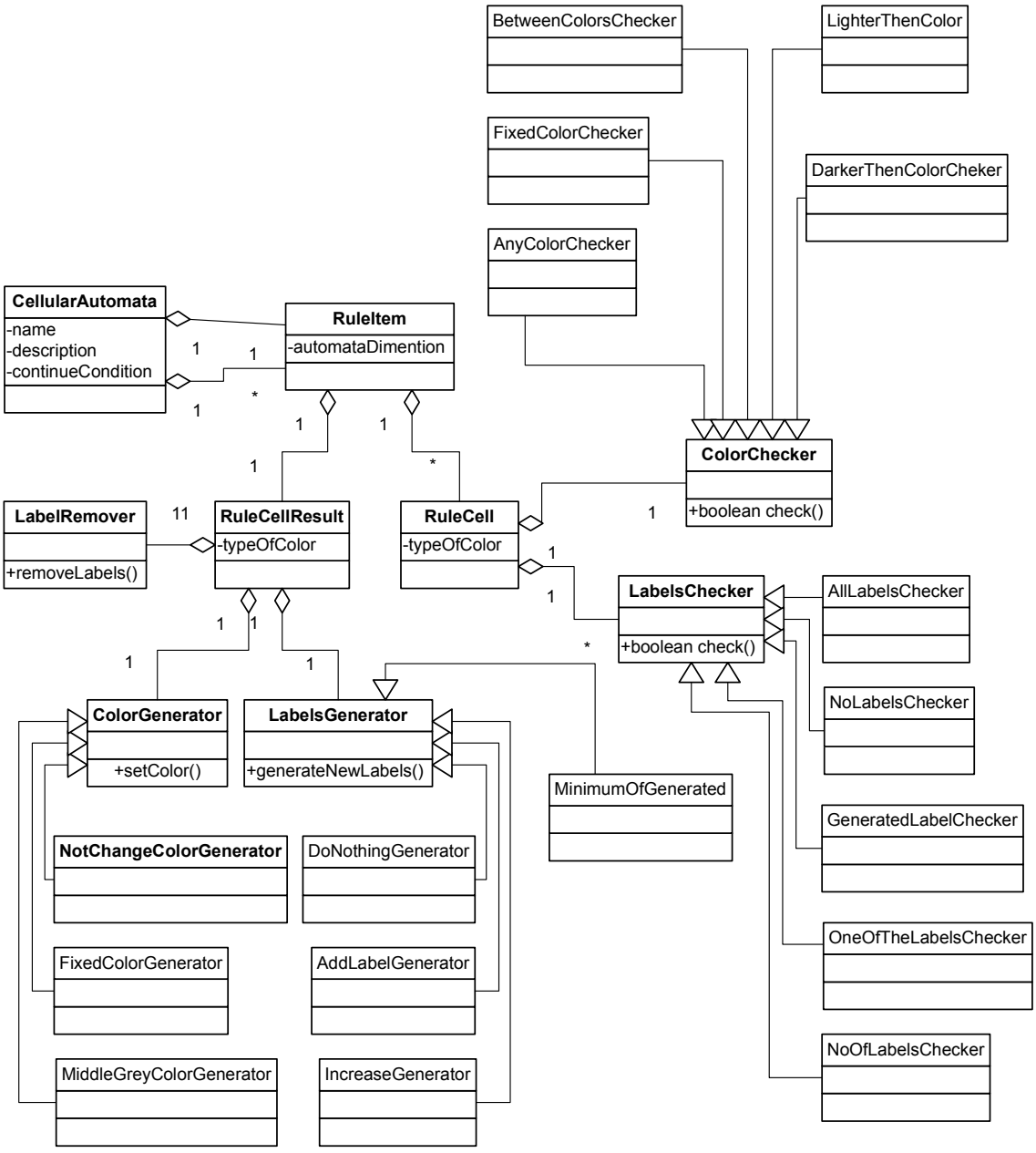


Рис. 27. Диаграмма классов клеточного автомата моделирующей программы

В табл. 1 представлено описание классов блока клеточного автомата.

Таблица 1. Описание классов блока клеточного автомата

Класс/поле/метод	**Описание**
`CellularAutomata` – базовый класс, описывающий работу клеточного автомата	
`String name`	Имя клеточного автомата
`String description`	Описание клеточного автомата
`ContinueCondition continueCondition`	Условие продолжения работы автомата (автомат не будет останавливаться, пока выполняется условие)
`RuleItem defaultRuleItem`	Правило клеточного автомата по умолчанию
`List<RuleItem> ruleItems`	Правила клеточного автомата
`void initialize (AutomataSequence automataSequence)`	Инициализация клеточного автомата в процессе начала работы последовательности клеточных автоматов
`ProcessResult process (MatrixElement[][] source)`	Запуск клеточного автомата на поле, заданном матрицей состояний клеток
`ProcessResult processStep (MatrixElement[][] source)`	Запуск шага клеточного автомата на поле, заданном матрицей состояний клеток
`RuleItem` – класс, описывающий правило клеточного автомата	
`Map<Integer, Map<Integer, RuleCell>> cells`	Клетки локального пространства, на которых определены условия выполнения данного правила клеточного автомата
`RuleCellResult result`	Результат выполнения правила клеточного автомата
`void initialize (MatrixElement[][]`	Инициализация правила клеточного автомата в процессе начала работы клеточного автомата

source)	
void process (MatrixElement[][] source, int x, int y)	Запуск правила клеточного автомата для клетки с координатами (x, y) поля, заданного матрицей состояний клеток
RuleCell – класс, описывающий условия, наложенные на клетку, в рамках правила клеточного автомата	
TypeOfColors colorTypeOf	Определяет совокупность цветов, допустимых в клетке
ColorChecker colorChecker	Объект для проверки цвета клетки
List<LabelChecker> labelCheckers	Объекты для проверки меток клетки
boolean process (MatrixElement[][] source, int x, int y)	Проверяет условия цвета и меток клетки правила клеточного автомата для клетки с координатами (x, y)
ColorChecker – абстрактный класс, проверяющий цвет клетки	
boolean check (MatrixElement[][] matrix, int x, int y)	Проверяет цвет клетки с координатами (x, y). Метод определен в потомках класса
AnyColorChecker – класс, проверяющий цвет клетки: принимает любой цвет клетки	
FixedColorChecker – класс, проверяющий цвет клетки: принимает фиксированный цвет клетки	
Color color	Цвет для проверки
BetweenColorsChecker – класс, проверяющий цвет клетки: принимает цвет клетки между двумя заданными	
Color firstColor	Первый цвет промежутка
Color secondColor	Второй цвет промежутка

DarkerThenColorCheker – класс, проверяющий цвет клетки: принимает цвет клетки темнее заданного	
`Color color`	Цвет для проверки
LighterThenColor – класс, проверяющий цвет клетки: принимает цвет клетки светлее заданного	
`Color color`	Цвет для проверки
LabelChecker – абстрактный класс, проверяющий метки клетки	
`boolean check (MatrixElement[][] matrix, int x, int y)`	Проверяет метки клетки с координатами (x, y). Метод определен в потомках класса
AllLabelsChecker – класс, проверяющий метки клетки: все метки должны присутствовать	
List<Labels> labelsToChecks	Метки для проверки
NoLabelsChecker – класс, проверяющий метки клетки: ни одной метки не должно быть	
GeneratedLabelChecker – класс, проверяющий метки клетки: должна присутствовать сгенерированная метка	
OneOfTheLabelsChecker – класс, проверяющий метки клетки: одна из меток должна присутствовать	
List<Labels> labelsToChecks	Метки для проверки
NoOfLabelsChecker – класс, проверяющий метки клетки: ни одна из списка метка не должна присутствовать	
List<Labels> labelsToChecks	Метки для проверки
RuleCellResult – класс, описывающий действия по определению цвета и меток клеток в рамках правила клеточного автомата	

TypeOfColors colorTypeOf	Определяет совокупность цветов, генерируемых в клетке
ColorGenerator colorGenerator	Объект для определения цвета клетки
LabelGenerator labelGenerator	Объект для определения меток клетки
LabelRemover labelRemover	Объект для удаления меток клетки
void initialize (RuleItem ruleItem, MatrixElement[][] source)	Инициализирует генерацию цвета и меток клеток
void process(int x, int y)	Выполняет правило генерации цвета и меток для клетки с координатами (x, y) клеточного автомата
ColorGenerator – абстрактный класс для генерации цвета клетки	
void initialize (MatrixElement[][] matrix)	Инициализирует генерацию цвета клеток
void setColor(int x, int y)	Определяет и выставляет цвет клетки с координатами (x, y). Метод определен в потомках класса
NotChangeColorGenerator – класс для генерации цвета клетки: не изменяет цвет клетки	
FixedColorGenerator – класс для генерации цвета клетки: выставляет заданный цвет клетки	
Color color	Цвет для генерации
MiddleGreyColorGenerator – класс для генерации цвета клетки: выставляет средний арифметический цвет клетки	
LabelGenerator – абстрактный класс для генерации меток клетки	
void initialize	Инициализирует генерацию меток клеток

(RuleItem ruleItem, MatrixElement[][] matrix)	
void run(int x, int y)	Определяет метки клетки с координатами (x, y). Метод определен в потомках класса
DoNothingGenerator – класс для генерации меток клетки: не добавляет метки	
AddLabelGenerator – класс для генерации меток клетки: добавляет метки к клетке	
List<Labels> labels	Метки для добавления
IncreaseGenerator – класс для генерации меток клетки: выставляет метку, следующую из неиспользованных	
MinimumOfGenerated – класс для генерации меток клетки: выставляет метку, минимальную из соседних	
LabelRemover – абстрактный класс для удаления меток клетки	
void initialize (MatrixElement[][] source)	Инициализирует удаление меток клеток
void removeLabels(int x, int y)	Удаляет метки из клетки с координатами (x, y)

4.1.2. Реализация последовательности клеточных автоматов

Моделирующая программа необходима для запуска последовательности клеточных автоматов. Реализация этого процесса предполагает наличие возможности запуска всей последовательности в целом, запуска отдельного клеточного автомата и шага клеточного автомата. Кроме клеточных автоматов в последовательности могут быть включены другие элементы: элемент разделения изображения текста на изображения отдельных символов, элемент проверки условий и перехода по данному условию на

другой элемент последовательности, элемент, который выставляет метку на заданную клетку (начальное условие), элемент, фиксирующий результат. Диаграмма классов последовательности клеточных автоматов показана на рис. 28.

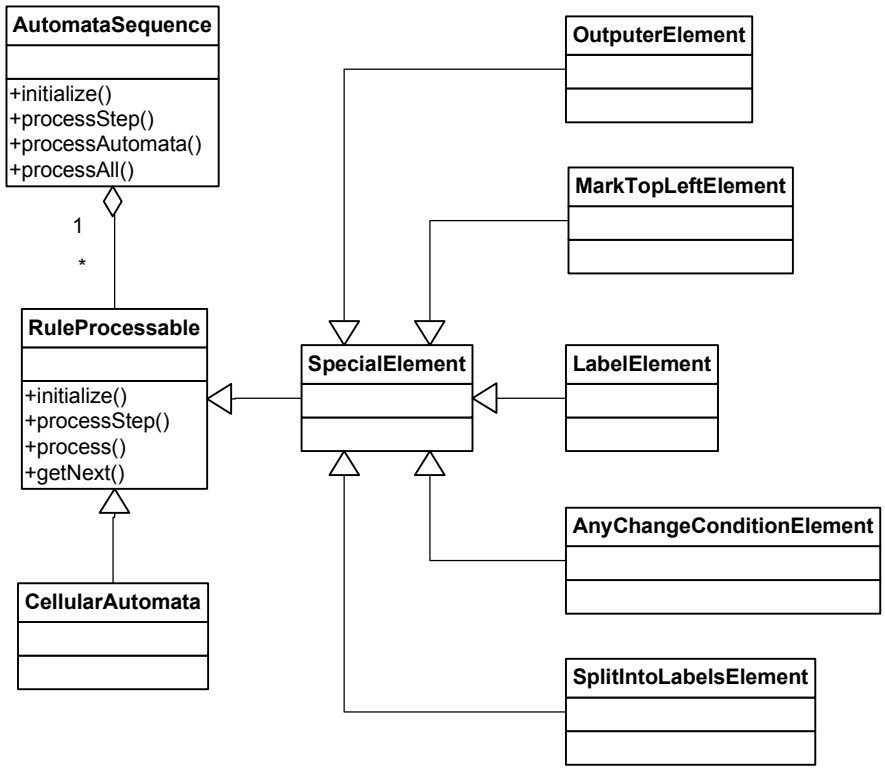


Рис. 28. Диаграмма классов последовательности клеточных автоматов моделирующей программы

В табл. 2 представлено описание классов блока последовательности клеточных автоматов.

Таблица 2. Описание классов блока последовательности клеточных автоматов

Класс/поле/метод	**Описание**
`AutomataSequence` – базовый класс, описывающий работу последовательности клеточных автоматов	
`List<RuleProcessable>`	Элементы последовательности клеточных автоматов

elements	
void addRule(RuleProcessable element)	Добавляет элемент в последовательность
RuleProcessable removeRule (int index)	Удаляет элемент из последовательности
void initializeProcessing (MatrixElement[][] matrixElements)	Инициализация работы последовательности клеточных автоматов
void processStep()	Запуск шага работы последовательности клеточных автоматов
void processAutomata()	Запуск клеточного автомата из последовательности
void processAll()	Запуск последовательности клеточных автоматов
RuleProcessable – абстрактный класс, описывающий работу элемента последовательности	
void initialize (AutomataSequence automataSequence)	Инициализация работы элемента последовательности. Метод определен в потомках класса
ProcessResult process (MatrixElement[][] source)	Запуск элемента последовательности. Метод определен в потомках класса
ProcessResult processStep (MatrixElement[][] source)	Запуск шага элемента последовательности. Метод определен в потомках класса
SpecialElement – абстрактный класс, описывающий работу специального элемента последовательности	
LabelElement – класс, определяющий метку в последовательности, определенный	

для условных элементов	
`AnyChangeConditionElement` – класс – условный элемент, переводящий работу последовательности на метку при условии выполнения действия в одном из клеточных автоматов между меткой и условием	
`LabelElement selectedLabel`	Метка для перехода
`SplitIntoLabelsElement` – класс, разделяет матрицу клеток на несколько матриц в зависимости от выставленных в них меток	
`List<Labels> labels`	Метка, учитывающиеся в разделении матрицы
`boolean generatedLabels`	Учитываются ли сгенерированные метки
`MarkTopLeftElement` – класс, метит верхнюю левую черную клетку в матрице	
`List<Labels> labels`	Метки, выставляемые в соответствии с правилами класса
`OutputerElement` – класс, выводит результат работы последовательности на экран	

4.1.3. Реализация блока обучения и классификации

В моделирующей программе блок обучения и классификации основан на статистическом методе: на этапе обучения система накапливает знания о признаках символов, а на этапе классификации она выбирает требуемый символ на основе этих знаний по выделенным признакам. Под признаками в данном случае понимается наличие у символа пересечений, концов и петель и их взаимное расположение.

Благодаря продуманной архитектуре, в программе предусмотрена возможность дополнения существующего инструмента классификации другими. Например, возможно использовать алгоритмы нейронных сетей или алгоритмы, базирующиеся на клеточных автоматах.

Диаграмма классов блока обучения и классификации показана на рис. 29.

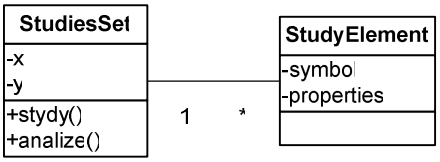


Рис. 29. Диаграмма классов блока обучения и классификации моделирующей программы

В табл.3 представлено описание классов блока обучения и классификации.

Таблица 3. Описание классов блока обучения и классификации

Класс/поле/метод	**Описание**
`StudiesSet` – **базовый класс, описывающий работу блока обучения и распознавания**	
`List<StudiesElement> elements`	Элементы обучения системы
`int x`	Индекс дробления по горизонтали
`int y`	Индекс дробления по вертикали
`void study(MatrixElement[][] matrix, List<Labels> acceptLabels, String text)`	Обучение системы на выделенных признаках, заданных матрицей клеток и списком распознаваемых меток, а также текст, соответствующий этим признакам
`List<AnalizeResult> analize (MatrixElement[][] matrix, List<Labels> acceptLabels)`	Распознавание признаков, заданных матрицей клеток и списком распознаваемых меток. Возвращает результат анализа, отсортированный по вероятности
`StudyElement` – класс – элемент обучения системы	
`Map<Integer, Map<Integer,`	Частота представления признака относительно меток и координат

`Map<Integer, Integer>>> numbers`	
`int getNumbers(int l, int x, int y)`	Возвращает частоту признаков относительно координат и метки

4.2. Описание работы моделирующей программы

Моделирующая программа построена на постраничной основе, где страницы с заданной функциональностью идут последовательно одна за другой. Всего в программе пять страниц: загрузка изображения, создание и изменение правил клеточных автоматов, комбинирования клеточных автоматов в последовательности, запуска последовательности и распознавания текста относительно данных, полученных в результате работы последовательности клеточных автоматов.

4.2.1. Страница загрузки изображения

Загрузка изображения происходит на первой странице. Изображение является входным данным для последовательности клеточных автоматов.

На данной странице для удобства создана возможность увеличения или уменьшения изображения.

На рис. 30 представлен снимок данной страницы.

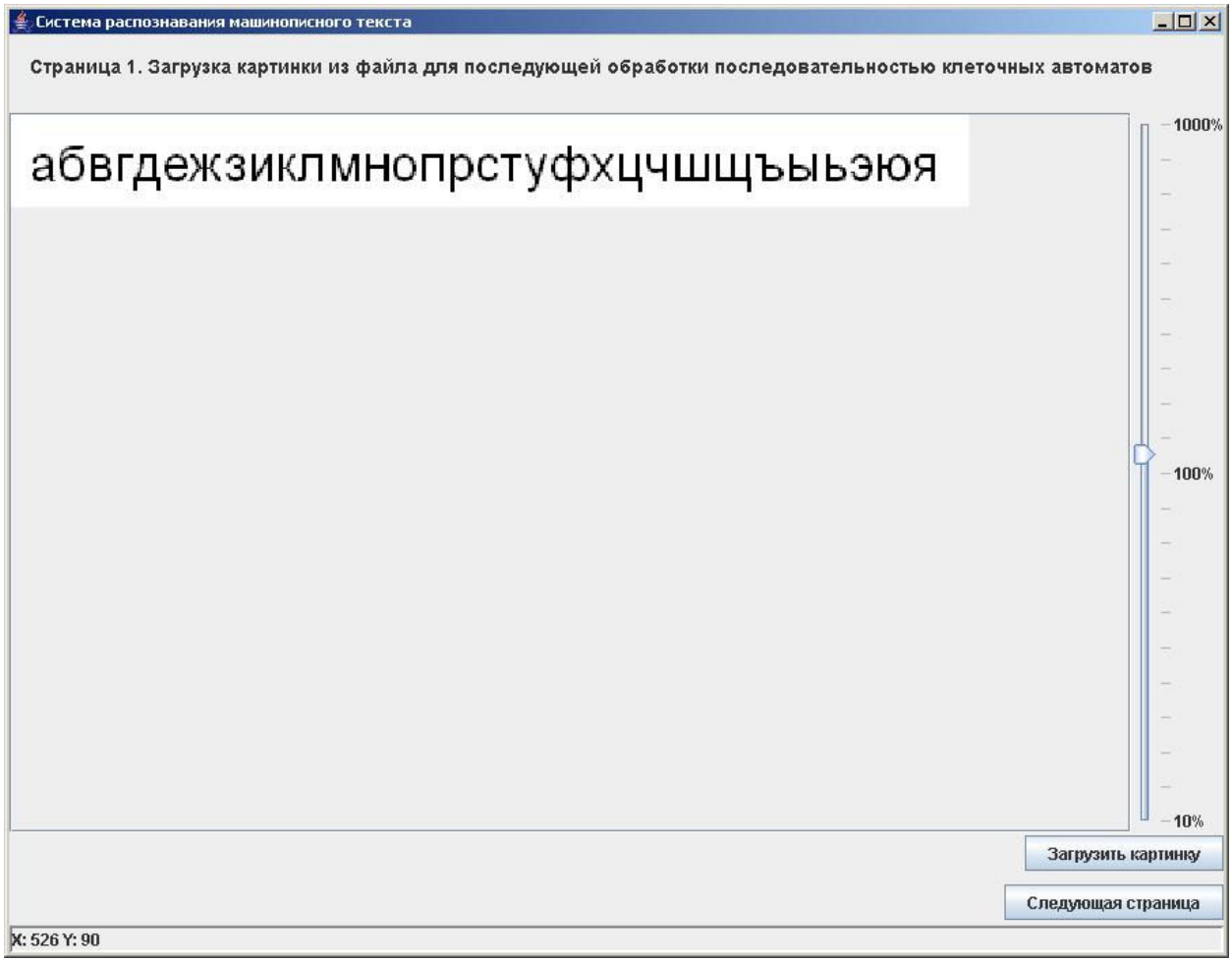


Рис. 30. Снимок страницы загрузки изображения

4.2.2. Страница создания и изменения правил клеточных автоматов

На второй странице возможно создание клеточных автоматов и их правил. Список правил отображается в правой части окна программы, из которого можно выбирать конкретное правило для редактирования в основной части окна.

На рис. 31 представлен снимок страницы работы с правилами автоматов.

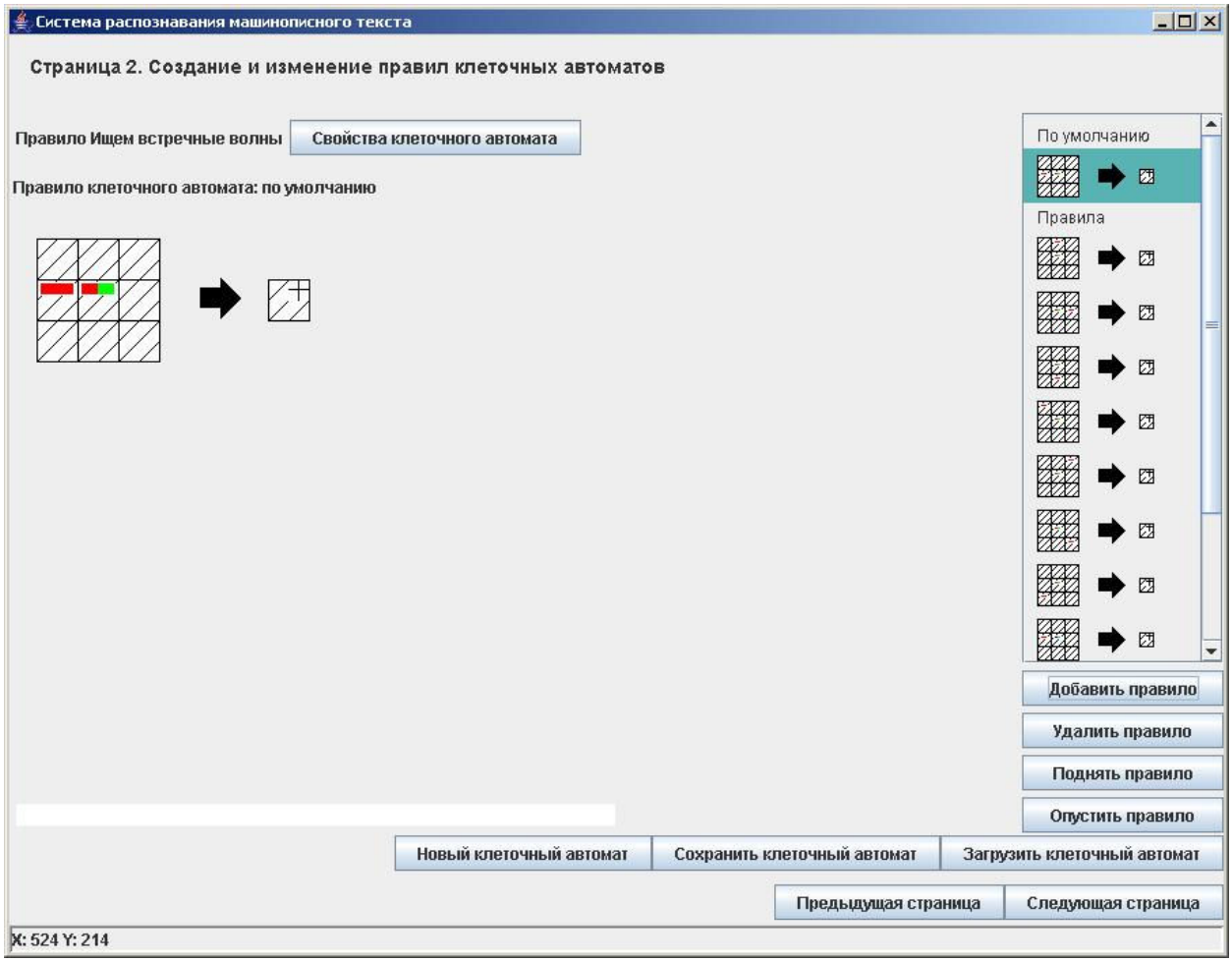


Рис. 31. Снимок страницы создания и изменения правил клеточных автоматов

4.2.3. Страница комбинирования клеточных автоматов в последовательности

Страница комбинирования клеточных автоматов позволяет составлять последовательности запуска клеточных автоматов и определять дополнительную логику запуска: непоследовательные переходы к следующим автоматам по условию, разбиение изображения, определение начального условия для работы клеточных автоматов.

На рис. 32 представлен снимок страницы комбинирования клеточных автоматов в последовательности.

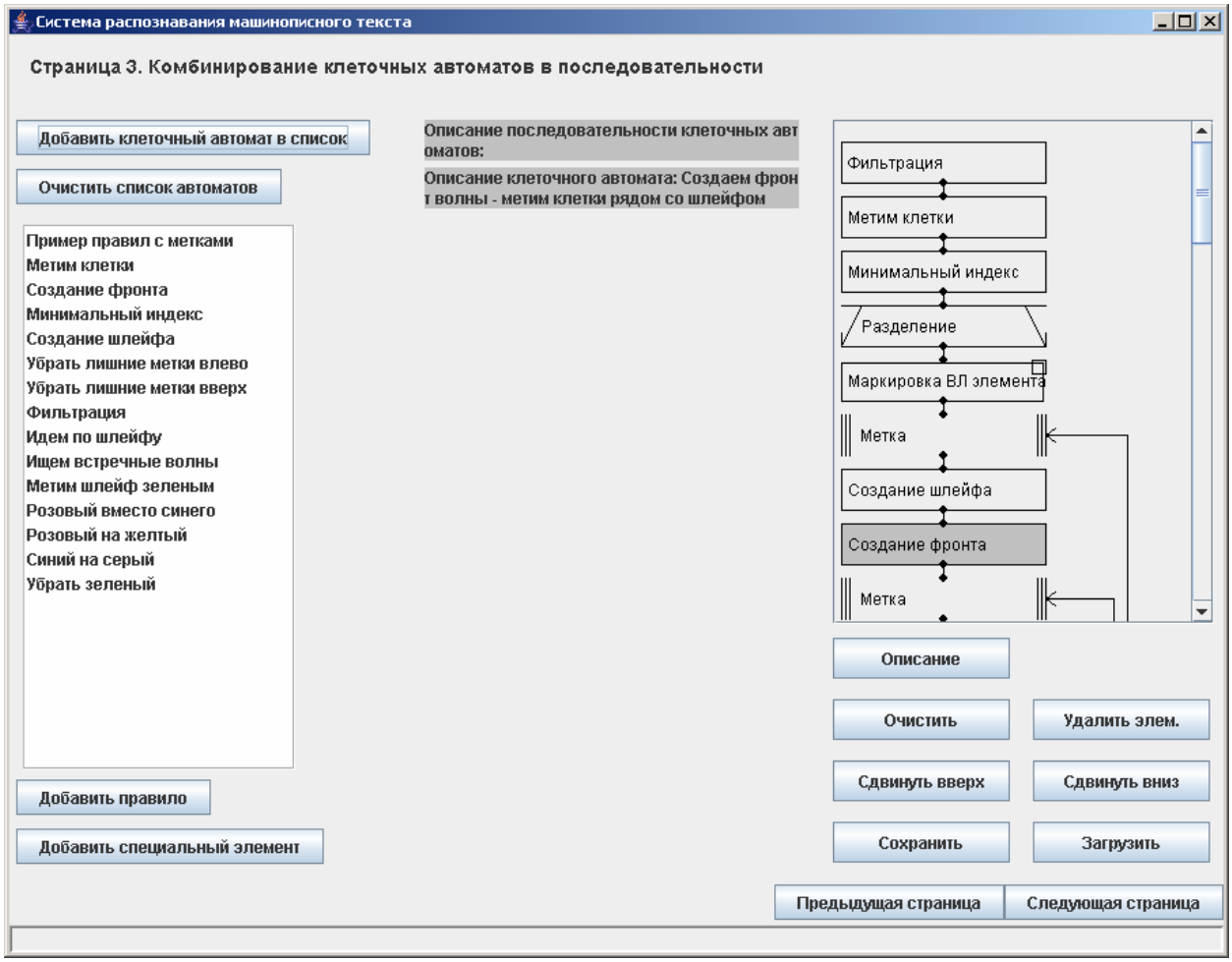


Рис. 32. Снимок страницы комбинирования клеточных автоматов в последовательности

4.2.4. Страница запуска последовательности клеточных автоматов

Четвертая страница программы служит для запуска последовательности клеточных автоматов. После загрузки последовательности из памяти программы или из файла последовательность можно запускать на выполнение. При этом доступны три режима работы: полный запуск всей последовательности, запуск автомата на выполнение и запуск шага автомата. На каждом шаге в основном окне можно просмотреть предварительные результаты работы.

На рис. 33 представлен снимок страницы запуска последовательности клеточных автоматов.

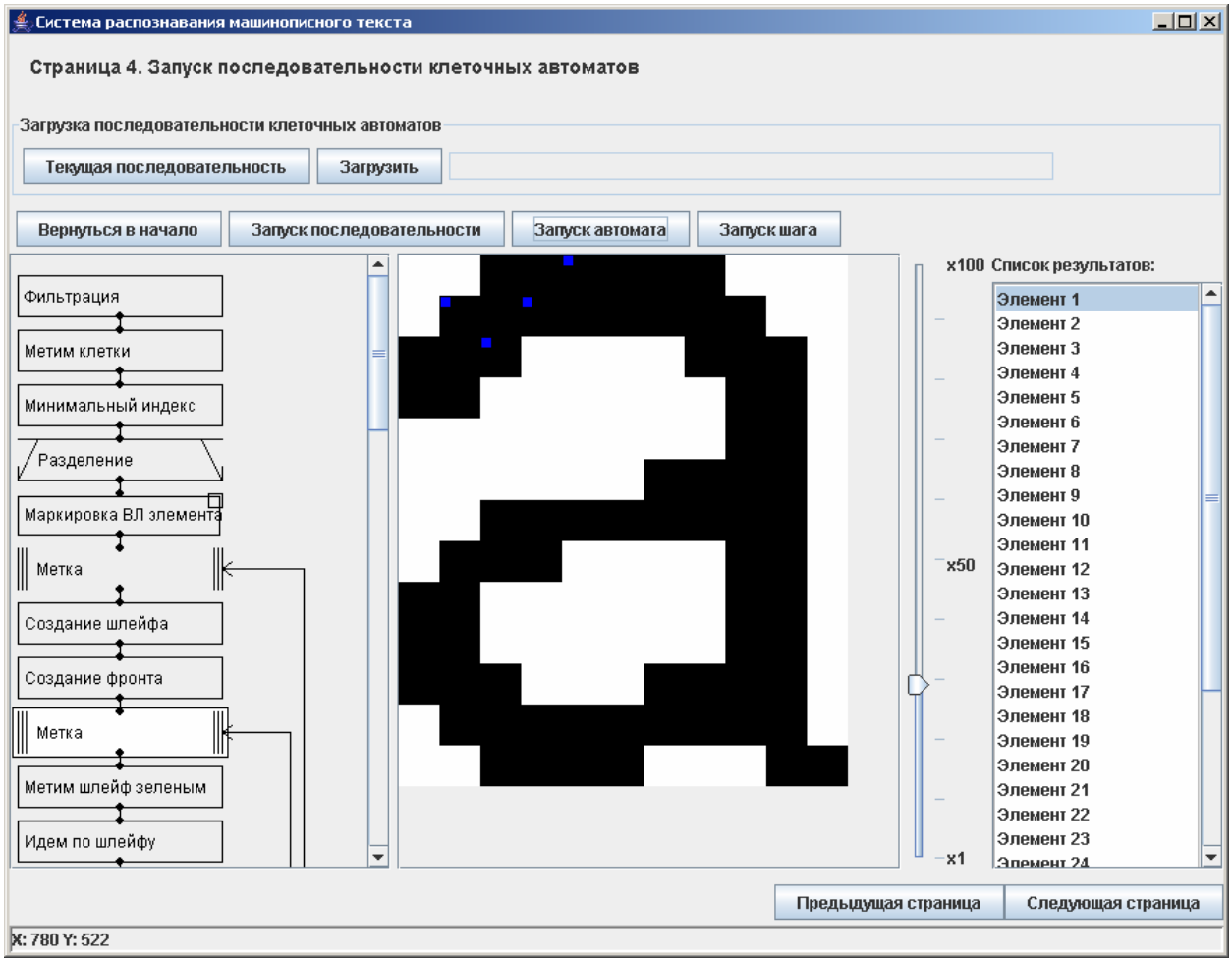


Рис. 33. Моментальный снимок страницы запуска последовательности клеточных автоматов

4.2.5. Страница распознавания текста

Страница распознавания использует данные, полученные в результате запуска последовательности клеточных автоматов, для обучения и распознавания символов текста. Для этого используются метки, которыми отмечались признаки символов, и метод распознавания, заданный в программе.

На рис. 34 представлен снимок страницы распознавания текста.

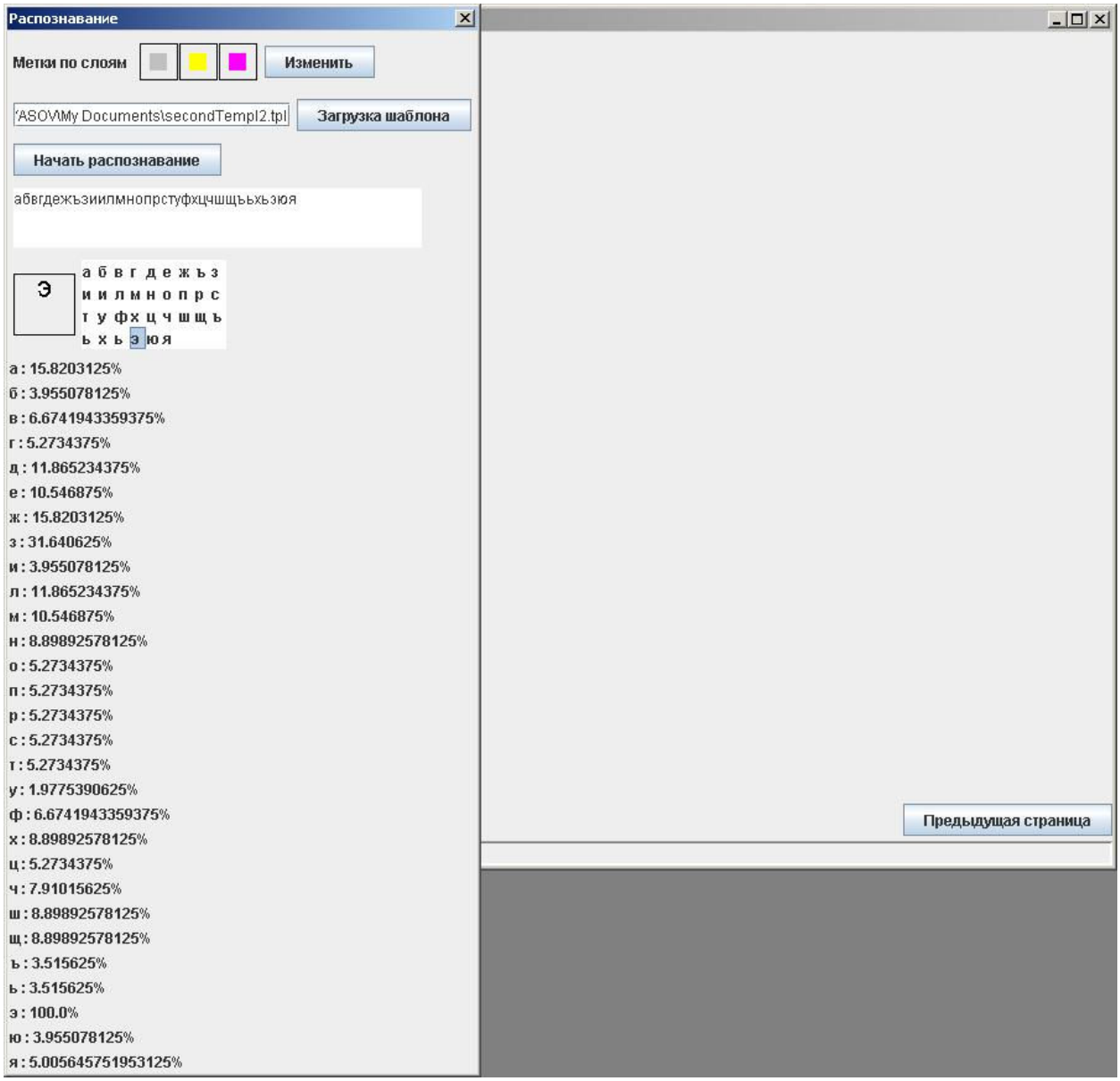


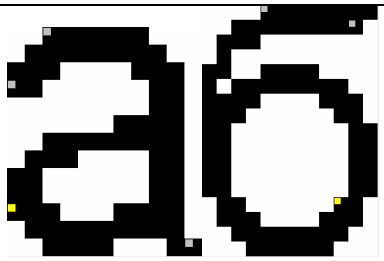
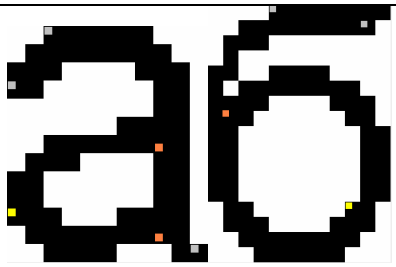
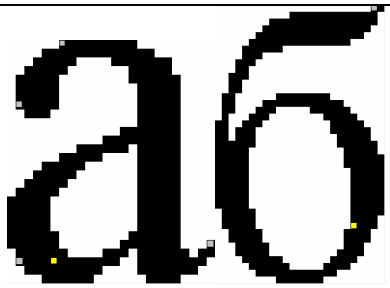
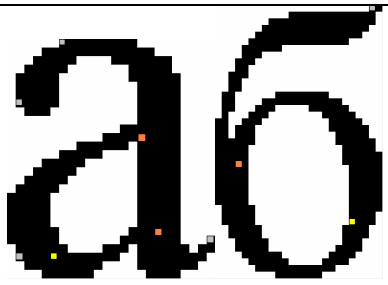
Рис. 34. Снимок страницы распознавания текста

4.3. Характеристики моделирующей программы

Программа позволяет реализовывать различные алгоритмы, основанные на клеточных автоматах и клеточных автоматах с метками, с дальнейшим запуском и проверкой работоспособности. В частности, программа позволяет выделить признаки символов из их изображений и классифицировать их.

На основе моделирующей программы были исследованы разработанные алгоритмы на основе клеточных автоматов для разных шрифтов символов. В табл. 4 представлен отрывок данного исследования.

Таблица 4. Отрывок исследования работы алготмов выделения признаков символов для разных шрифтов

Характеристики текста	**Первый алгоритм выделения признаков символов**	**Второй алгоритм выделения признаков символов**
Символы небольшого размера шрифта *Arial*		
Символы большого размера шрифта *Arial*		
Символы большого размера шрифта *New Times Roman*		

Результаты показали устойчивость работы алгоритма для разных размеров одного и того же шрифта. В то же время для разных шрифтов алгоритмы выделяют несколько отличные комбинации признаков и их расположений, что подтверждает необходимость учета шрифта при использовании систем *OCR* на основе данных алгоритмов.

Разработанная моделирующая программа построена с ориентацией на расширяемость: учтена возможность добавления новых элементов, необходимых для проведения исследований.

Производительность системы, во многом, зависит от заданной последовательности клеточных правил и напрямую вычисляется из количества шагов, выполняемых клеточными автоматами. Основное время работы программы занимает обход поля для реализации работы клеточных автоматов. Для оптимизации алгоритма обхода следует учитывать принцип локальности правил и реализовывать принцип параллельного вычисления.

Язык программирования, использовавшийся для написания программы, *Java*. Он выбран с учетом возможности запуска программы на любой платформе, а также из Интернет-браузера на основе технологии *Applet*.

Скоростные характеристики программы для первого алгоритма представлены в табл. 5.

Таблица 5. Скоростные характеристики моделирующей программы

Слова для обработки последовательностью	Размер изображения в пикселях	Количество символов	Количество итераций автоматов	Время работы, с
«Документирование»	236 x 30	16	10947	6
Алфавит кроме букв «й», «ё» и «ы»	454 x 44	30	20250	13
Текст	561 x 56	76	46006	16

Невысокие характеристики работы определяются моделирующим характером разработанной программы. Используемые алгоритмы требуют оптимизации и доработки.

4.4. Пример работы алгоритмов распознавания текста

В данном разделе приведен пример обучения системы и распознавания текста.

4.4.1. Обучение системы

Обучение системы состоит в запуске алгоритмов выделения признаков символов на специально созданном примере. Этот пример должен содержать, по возможности, все обрабатываемые системой символы. Наиболее оптимально проводить обучение на изображении алфавита.

После обработки последовательностью клеточных автоматов, в каждом изображении символа алфавита будет выделены признаки символов. Пользователь программы ассоциирует набор признаков конкретному символу, и данная ассоциация сохраняется. В дальнейшем информацию о соответствии признаков определенным символам можно будет использовать в процессе распознавания текстов, написанных теми же шрифтами, что и обучающий текст.

На рис. 35 приведен пример алфавита, на котором проводится обучение в примере. Данное изображение загружается на первой странице программы.

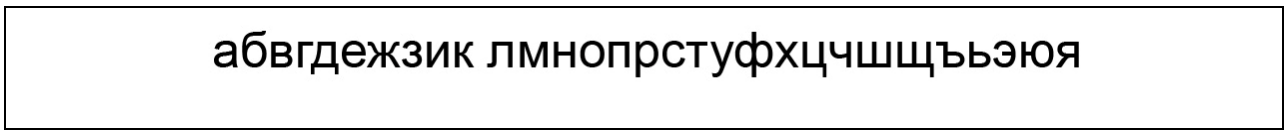


Рис. 35. Пример изображения алфавита для обучения системы распознавания

После запуска последовательности клеточных автоматов, которые преобразуют изображение в черно-белое, разделяют изображение текста на изображения отдельных символов и выделяют признаки изображений (используется первый алгоритм выделения признаков символов), в

результате для каждого символа появляется набор признаков. На рис. 36 схематично изображены признаки первых четырех букв алфавита.

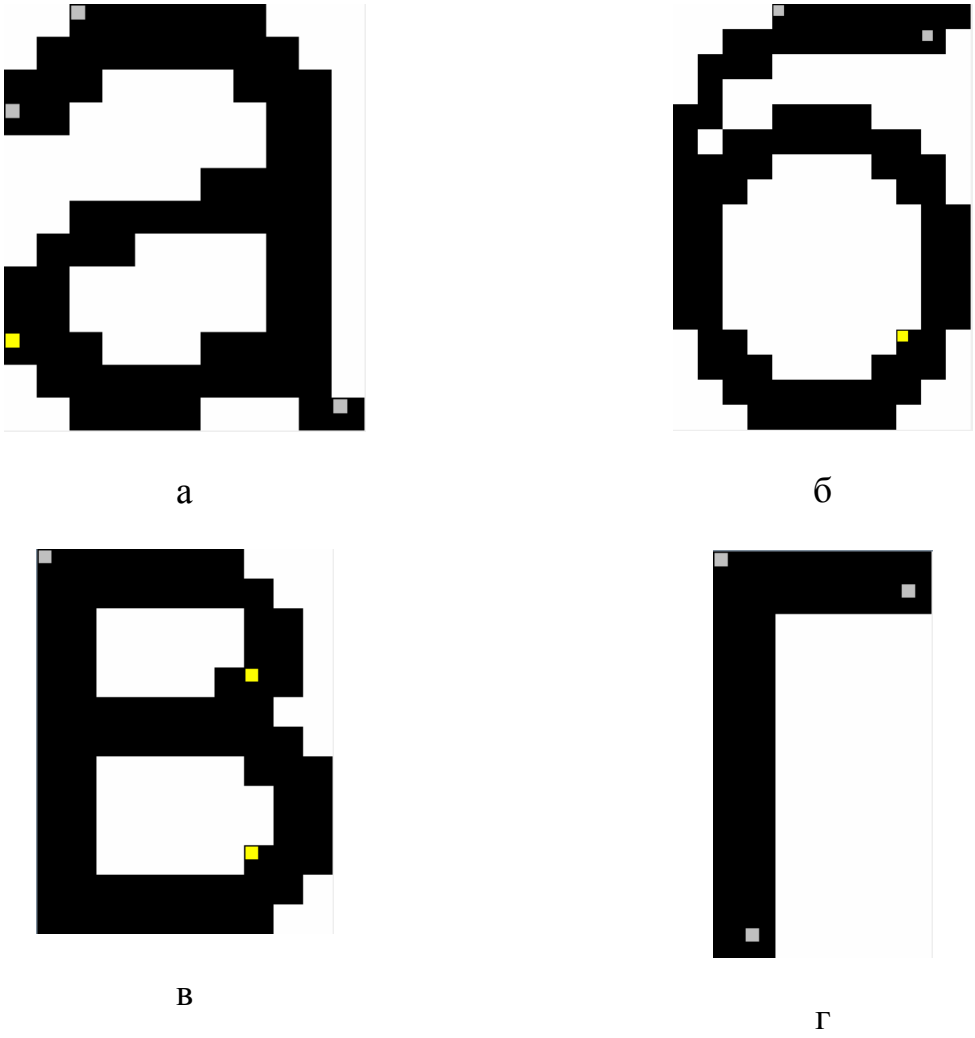


Рис. 36. Признаки (расположение концов и петель) четырех символов алфавита

После выделения признаков, им в соответствие необходимо выставить символы. Для этого, необходимо выбрать, какие метки в изображении отвечают за признаки символов, как показано на рис. 37.

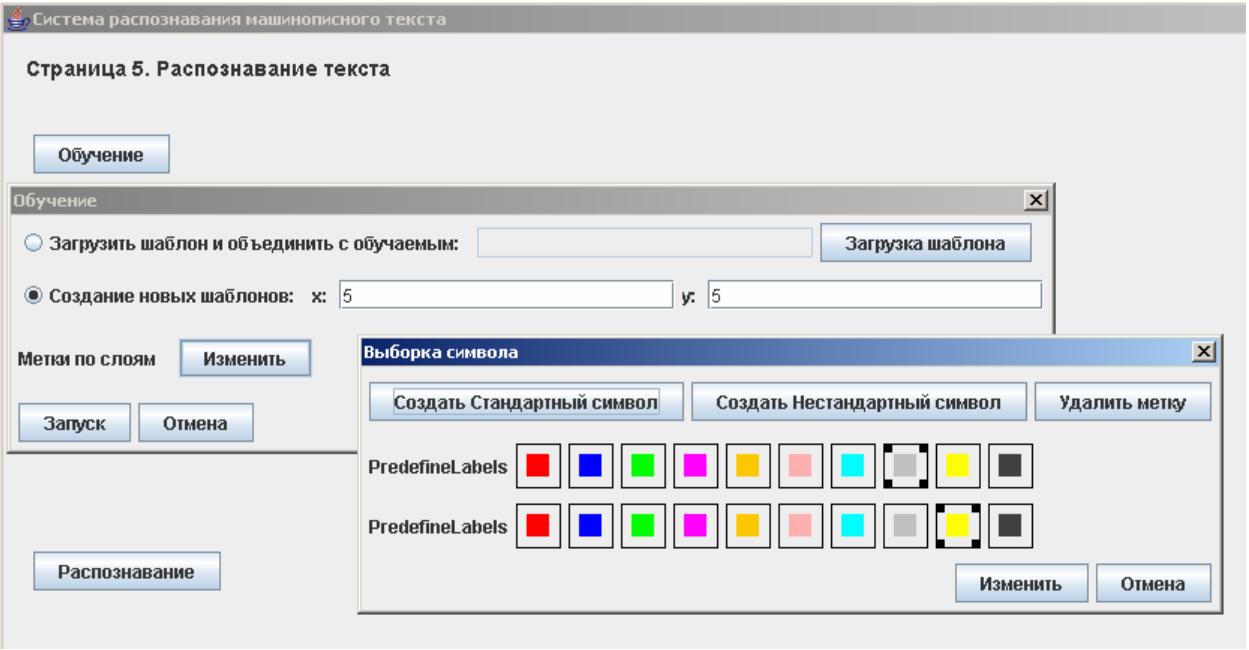


Рис. 37. Выбор меток – признаков символов, на основе которых проводится обучение

После определения меток обучение запускается кнопкой `Запуск`. После этого появляется специальное окно, в котором пользователь вводит символ, соответствующий изображению, как показано на рис. 38.

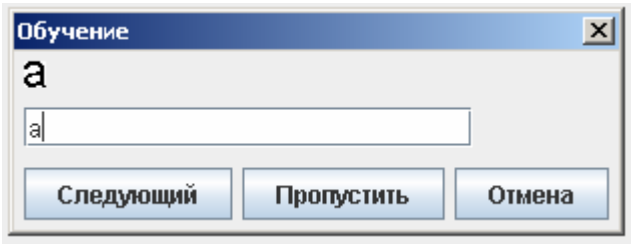


Рис. 38. Диалоговое окно моделирующей программы блока обучения

Как только все символы определены пользователем, данные можно сохранить в файл и затем использовать в блоке распознавания текстов.

4.4.2. Распознавание текста

Последовательность действий в моделирующей программе, соответствующая распознаванию текста практически аналогична действиям, необходимым для обучения: необходим запуск последовательности

клеточных автоматов для преобразования изображения в черно-белое, разделения изображения текста на изображения отдельных символов и выделения признаков символов из изображений.

Для примера взято изображение слова «документирование», как показано на рис. 39.

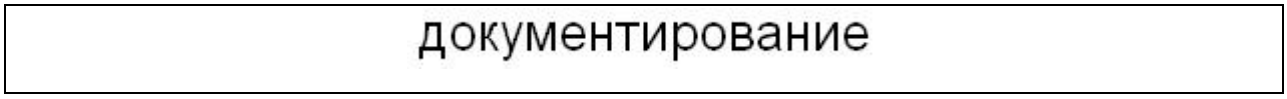
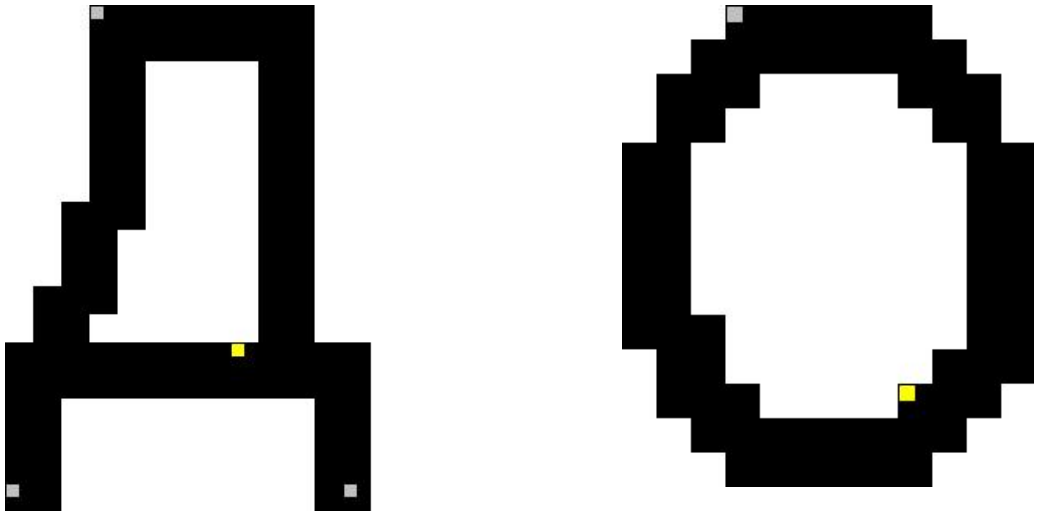


Рис. 39. Изображение распознаваемого слова «документирование»

После завершения работы последовательности клеточных автоматов (четвертая страница моделирующей программы, разд. 4.2.4) из изображений символов слова будет выделены признаки. Признаки первых двух букв слова представлены на рис. 40.



а б

Рис. 40. Признаки (расположение концов и петель) букв: а – Д, б – О

После выделения признаков символов на пятой странице моделирующей программы (разд. 4.2.5) запускается распознавание текста на основе этих признаков. Для этого аналогично процессу обучения выбираются метки, отвечающие за признаки символов, и загружается файл шаблона, который содержит информацию обучения на алфавите.

Результатом распознавания будет текст, каждый символ которого определен с заданной точностью на основе статистического метода. Результат распознавания слова «документирование» представлен на рис. 41.

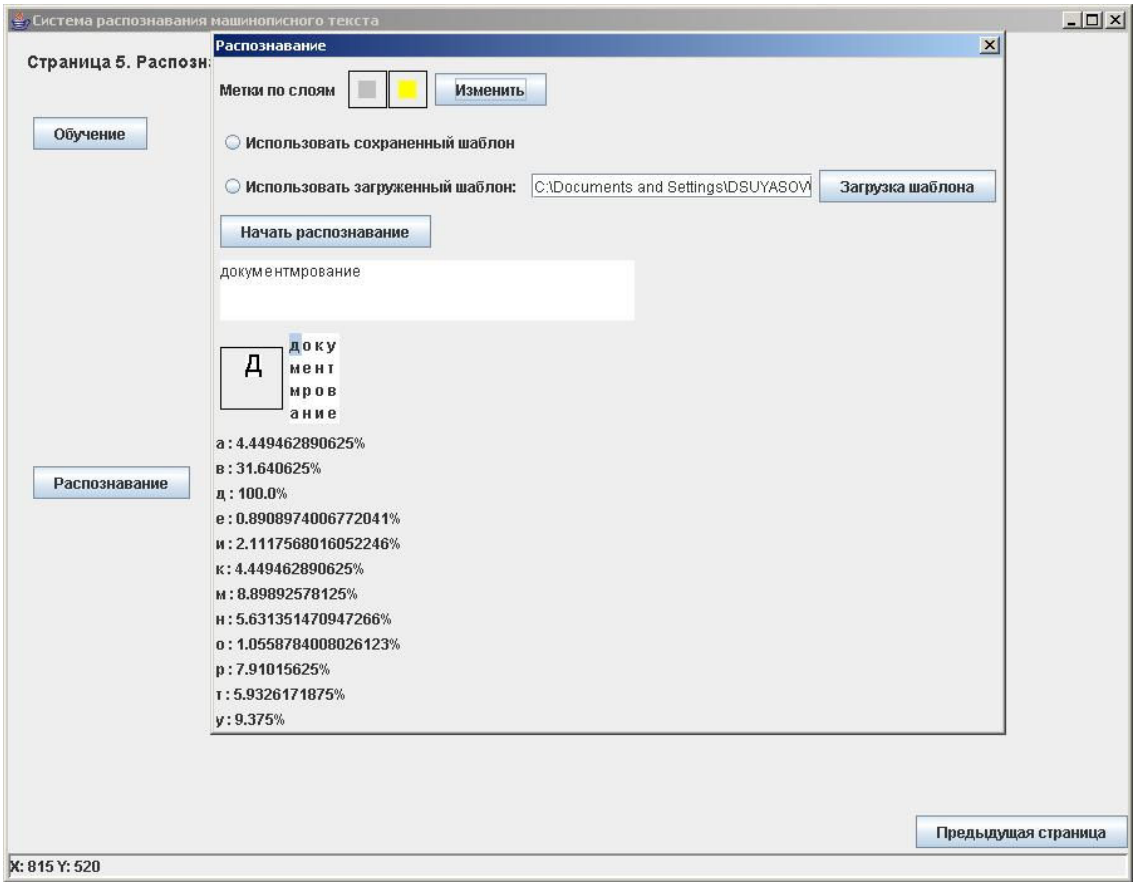


Рис. 41. Окно распознавания текста моделирующей программы с результатами распознавания слова «документирование»

Как видно из примера результат содержит ошибку в девятом символе, что означает недостаток информации для корректного распознавания. Более точный результат может быть получен с помощью второго алгоритма выделения признаков символов.

ГЛАВА 5. СРАВНЕНИЕ С ДРУГИМИ МЕТОДАМИ И ВЫВОДЫ ПО РАБОТЕ

5.1. Сравнение с другими методами

Основной задачей разработанных алгоритмов в процессе распознавания текста – это обработка изображения и выделение признаков. Большинство подобных систем являются коммерческими продуктами, и информация об используемых алгоритмах не размещается в публичном доступе. Поэтому сравнение с используемыми методами невозможно, в виду закрытости информации.

В мире представлено несколько открытых программных средств по распознаванию текста. К сожалению, практически все из них характеризуются отсутствием документации к используемым методам и алгоритмам. Большинство программ находится в начальной стадии разработки (*ImageTextEditor* [23], *OpenOCR* [24]), некоторые специально оптимизированы под *Unix* системы (*OCRAD* [25], *ImageTextEditor*), часть использует алгоритмы, оптимизированные под конкретные языки (*Kognition* для немецкого языка [26], *Tesseract* для английского языка [27]).

Наиболее проработанным открытым программным средством является *Tesseract*. Эта система разрабатывалась с 1985 года вначале *Hewlett-Packard*, затем университетом Невады в Лас-Вегасе (University of Nevada in Las Vegas) и дорабатывалась компанией *Google*, которая в августе 2006 года открыла исходные тексты программы. Программа хорошо распознает тексты на основе шрифтов без засечек, для чего использует комплекс алгоритмов, оптимизированных для символов английского алфавита. В сравнении с *Tesseract* разработанные в данной работе алгоритмы не зависят от распознаваемого языка и служат для исследования поведения клеточных автоматов в системах распознавания текста. *Tesseract* работает быстрее

разработанной автоматом моделирующей программы, но это лишь подтверждает исследовательский характер данной работы.

В сравнении с нейронными сетями, алгоритмы, представленные в данной работе, выполняют другую задачу, а именно выделяют характеристики символов из их изображений. В то время как нейронные сети используются для классификации выделенных признаков.

Нейронные сети могут использоваться в комплексе с разработанными методами выделения признаков символов. В этом случае они будут решать задачи классификации признаков, выделенных алгоритмами на основе клеточных автоматов.

Реализация алгоритмов распознавания текста на основе клеточных автоматов была сравнена с реализацией системы распознавания текста на основе нейронных сетей, выполненной в курсовой работе автора. При этом если программа на основе нейронных сетей показала быстродействие в среднем большее по сравнению с программой на основе клеточных автоматов, то процент ошибок оказался ниже у алгоритмов на основе клеточных автоматов в два раза.

В табл. 6 представлена сравнительная характеристика работы разработанной моделирующей программы на основе клеточных автоматов с курсовой работой по распознаванию текста на основе нейронной сети и коммерческим продуктом *FineReader* [28] компании *ABBYY*.

Таблица 6. Сравнительная характеристика работы моделирующей программы на основе клеточных автоматов, курсовой работы на основе нейронных сетей и коммерческого продукта *FineReader*

Тестовые значения	Моделирующая программа		Курсовая работа		FineReader	
	Время работы, с.	Процент ошибок	Время работы, с.	Процент ошибок	Время работы, с.	Процент ошибок
Изображение слова «документирование», размеры: 236 x 30	6	4%	2	9%	1	0%
Изображение русского алфавита без букв «й», «ё» и «ы»	13	3%	4	11%	1	3%
Изображение русского текста (76 символов), размеры: 561 x 56	16	11%	5	17%	3	1%
Изображение страницы текста (742 символа)	83	12%	40	16%	7	2%

На основании приведенных в табл. 6 данных можно сделать несколько выводов:

1. Характеристика качества распознавания выше у моделирующей программы в сравнении с курсовой работой на основе нейронных сетей. Коммерческая программа *FineReader* превосходит по качеству распознавания все рассматриваемые.

2. По скорости работы превосходит коммерческая программа *FineReader*, затем курсовая программа на основе нейронных сетей и с

небольшим отставанием от последней находится моделирующая программа, разработанная автором работы.

Высокие показатели точности распознавания текста коммерческим продуктом *FineReader* определяются использованием комплекса средств по распознаванию, в том числе, активное использование словарей [4]. Также можно добавить, что *FineReader* является законченным продуктом, разрабатываемым на протяжении нескольких лет компанией *ABBYY*. Поэтому, представленное сравнение носит информативный характер и не может являться основанием для выводов относительно эффективности работы программ.

Невысокие характеристики скорости работы разработанной программы в качестве *OCR* системы определяется моделирующим характером программы. При разработке моделирующей программы основной задачей ставилось исследование алгоритмов на основе клеточных автоматов в системе распознавания текста с возможностью поэтапного, пошагового обзора результатов. Оптимизация алгоритмов распознавания значительно увеличит производительность моделирующей программы. Данная оптимизация возможна на основе нескольких принципов:

- введение обобщенных координат [29];
- исключение неиспользуемых частей полей, на которых работают клеточные автоматы;
- объединение точек изображения в клетки крупного размера и т.д.

5.2. Выводы по работе

Разработанные алгоритмы по обработке изображения, разделения изображения текста на изображения отдельных символов, а также алгоритмы выделения признаков символов позволяют выполнить часть задач процесса распознавания текста. Эти алгоритмы базируются на последовательностях

клеточных автоматов с метками, что дает им все преимущества использования клеточных автоматов.

1. Разработанные алгоритмы могут участвовать в параллельных вычислениях.

2. Алгоритмы выделяют уникальные признаки символов и позволяют на их основе реализовать дальнейшую классификацию.

3. Алгоритмы устойчивы к размерам символов, но зависят от используемого шрифта текста, в особенности, наличие засечек.

4. Алгоритмы основаны на простых правилах и не содержат сложных вычислений, в особенности с плавающей точкой.

Алгоритмы могут быть задействованы в комплексном решении задачи распознавания текста, где часть алгоритмов будет основана на другом принципе.

Алгоритмы реализованы и опробованы в разработанной моделирующей программе. Они показали работоспособность и возможность дальнейшего совершенствования.

Работа носит исследовательский характер. Основной целью исследования является определение перспектив клеточных автоматов в системе распознавания текста. Разработанные алгоритмы и их проверка в моделирующей программе показали возможность использования нового метода в выполнении задач распознавания текста.

Исследование принципов распознавания текста на основе клеточных автоматов и их последовательностей имеют несомненные перспективы, и будет автором продолжаться.

ЗАКЛЮЧЕНИЕ

В работе проведено исследование применимости клеточных автоматов в задаче распознавания текста, проанализирована каждая подзадача процесса распознавания и оценена возможность ее решения с помощью клеточных автоматов.

Для решения задачи распознавания введено понятие клеточных автоматов с метками и последовательностей клеточных автоматов. Клеточные автоматы с метками имеют общие черты с понятием памяти тьюринговых машин [20].

На основе введенных понятий разработаны алгоритмы по фильтрации изображения, сегментации изображения текста на отдельные изображения символов и выделение признаков символов.

Данные алгоритмы позволяют проводить параллельные вычисления на важных стадиях процесса распознавания текста. Кроме того, благодаря использованию механизма клеточных автоматов возможна аппаратная реализация алгоритмов.

Для реализации разработанных алгоритмов создана моделирующая программа. Она позволила по шагам оценить эффективность созданных алгоритмов и провести эксперименты по распознаванию текста.

ИСТОЧНИКИ

[1] *Нейман Дж.* Теория самовоспроизводящихся автоматов. Дж. Нейман. М.: Мир, 1971.

[2] *Wolfram S.* A New Kind of Science. Wolfram Media. Inc., 2002.

[3] *Колесников С.* Распознавание образов. Общие сведения /Газета "Компьютер-Информ". Программное обеспечение. http://www.ci.ru/

[4] *Травин А.* Технологии оптического распознавания текстов // Электронный офис. 1996. Ноябрь.

[5] *Хайкин С.* Нейронные сети: полный курс. М. : Вильямс, 2006.

[6] *Терехов С. А.* Лекции по теории и приложениям искусственных нейронных сетей. Лабораторотория Искусственных Нейронных Сетей НТО-2. Снежинск. ВНИИТФ.

[7] *Smith R.A.* Real-Time Language Recognition by One-Dimensional Cellular Automata // J. of Computer and System Sciences, v. 6 (1972), pp. 233–253.

[8] *Buchholz T., Klein A., Kutrib M.* Real-Time Language Recognition by Alternating Cellular Automata /IFIG Research Report 9904. 1999. March.

[9] *Тоффоли Т., Марголус Н.* Машины клеточных автоматов. М.: Мир, 1991.

[10] *Астафьев Г.Б., Короновский А.А., Храмов А.Е.*. Клеточные автоматы. Саратов: Колледж. 2003. http://cas.ssu.runnet.ru/sgnp/data/papers/Train/CellAutomat.pdf

[11] *Ulam S.* Random Processes and Transformations / Procedings Int. Congr. Mathem. 1952. №2, pp. 264–275.

[12] *Короновский А.А., Храмов А.Е., Анфиногентов В.Г.* Феноменологическая модель электронного потока с виртуальным катодом // Известия РАН. Сер. Физическая. 1999. Т.63, № 12, с. 2355–2362.

[13] *Shackleford B., Tanaka M., Carter R., Snider G.* Cellular and Cryptographic Applications: FPGA implementation of neighborhood-of-four cellular automata random number generators. ACM Press. 2002.

[14] *Wolfram S.* Cellular automation Fluids // J. Stat. Phys. 1986. Vol. 45, PP. 471–526.

[15] *Frish U. et al.* Lattice gas hydrodynamics in two and three dimensions // Complex Systems. 1987. Vol. 1, PP. 649–707.

[16] *Малинецкий Г.Г., Степанцев М.Е.* Моделирование движения толпы при помощи клеточных автоматов //Известия ВУЗов. Сер. “Прикладная нелинейная динамика”. 1997. Т. 5, с. 75–79.

[17] *Chopard B., Droz M.* Cellular automata model for heat conduction in a fluid // Physics Letters A. 1988. Vol. 126. N 8/9, PP. 476–480.

[18] *Наумов Л.А.* Разработка среды и библиотеки CAME&L для решения задач с использованием клеточных автоматов. СПбГУ ИТМО, 2003. http://is.ifmo.ru/papers/camel/

[19] *Gardner M.* The Fantastic Combinations of John Conway’s New Solitaire Game “Life”// Scientific American. 1970. №223, pp. 120–123.

[20] *Хопкрофт Д., Мотвани Р., Ульман Д.* Введение в теорию автоматов, языков и вычислений. М.: Вильямс. 2002.

[21] *Малинин В.В.* Распознавание образов на ЭВМ. ЦИТ СГГА, 2005.

[22] *Orovas C.* Cellular Associative Neural Networks for Pattern Recognition. University of York, 1999.

[23] *Проект* ImageTextEditor. http://imated.sourceforge.net

[24] *Проект* OpenOCR. http://www.oocr.org/

[25] *Проект* OCRAD. http://www.gnu.org/software/ocrad/ocrad.html

[26] *Проект* Kognition. http://sourceforge.net/projects/kognition/

[27] *Проект* Tesseract. http://sourceforge.net/projects/tesseract-ocr/

[28] *Optical* Character Recognition System ABBYY FineReader. http://www.abbyy.com/

[29] *Наумов Л.А.* Метод введения обобщенных координат и инструментальное средство для автоматизации проектирования программного обеспечения вычислительных экспериментов с использованием клеточных автоматов. Диссертация на соискание ученой степени канд. техн. наук. СПбГУ ИТМО, 2007. http://is.ifmo.ru/papers/_ln_Thesis.pdf

ПРИЛОЖЕНИЕ. Архитектура моделирующей программы

Моделирующая программа создана для исследований работы различных клеточных автоматов. Ее создание проводилось с учетом дополнения отдельных ее компонентов, и поэтому архитектура приложения основывается на модульном принципе.

В программе кроме блоков, отвечающих за реализацию клеточных автоматов и их последовательностей, включены вспомогательные элементы, упрощающие работу с ней. Эти элементы: блок записи и загрузки, блок реализации страниц пользовательского интерфейса, языковой блок.

П.1. Блок записи и загрузки

В программе присутствует возможность записи и загрузки различных сущностей: картинок, клеточных автоматов, последовательностей клеточных автоматов, результатов обучения системы. Основной принцип записи и загрузки – реализация интерфейса `XMLTransferable` всеми хранимыми в файлах элементами и наличие у них конструктора, который воспринимает запись в *XML*-формате. Диаграмма классов, реализующих данный принцип, показана на рис. П.1.

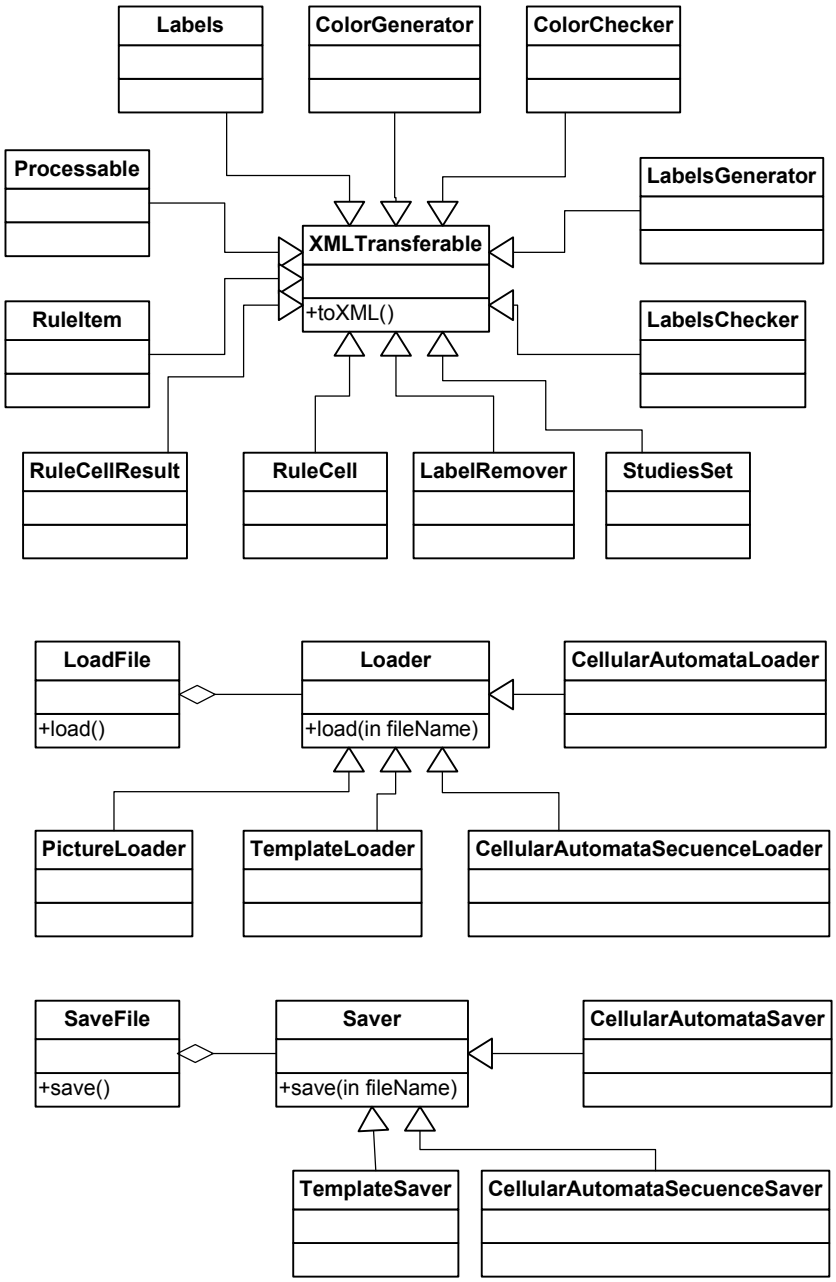


Рис. П.1. Диаграмма классов блока записи и загрузки

В таблице П1 представлено описание классов блока записи и загрузки.

Таблица П1. Описание классов блока записи и загрузки

Класс/поле/метод	**Описание**
`XMLTransferable` – интерфейс преобразования объектов в *XML* формат	
`void toXML(Document doc, Node rootElement)`	Метод преобразования объекта в *XML* формат. Реализован в классах и наследниках `Labels`, `ColorChecker`, `ColorGenerator`, `LabelsGenerator`, `LabelsChecher`, `StudiesSet`, `LabelRemover`, `Processable`, `RuleCell`, `RuleCellResult`, `RuleItem`.
`LoadFile` – класс для загрузки объекта из файла	
`Object load()`	Загружает объект из файла
`Loader` – интерфейс, для загрузки объекта определенного типа из файла	
`Object load(String fileName)`	Загружает объект определенного типа из файла с заданным именем. Метод реализован в потомках: `CellularAutomataLoader` – загружает сохраненные клеточные автоматы, `CellularAutomataSequenceLoader` – загружает последовательности клеточных автоматов, `PictureLoader` – загружает изображение для обработки, `TemplateLoader` – загружает обученный шаблон для распознавания.
`SaveFile` – класс для сохранения объекта в файл	
`boolean save()`	Сохраняет объект в файл
`Loader` – интерфейс, для сохранения объекта определенного типа в файл	
`boolean save(String fileName)`	Сохраняет объект определенного типа в файл с заданным именем. Метод реализован в потомках: `CellularAutomataSaver` – сохраняет клеточные автоматы, `CellularAutomataSequenceSaver` – сохраняет последовательности клеточных автоматов,

	`TemplateSaver` – сохраняет обученный шаблон

П.2. Блок реализации страниц пользовательского интерфейса

Пользовательский интерфейс основан на страницах, отвечающих своим задачам: страница загрузки картинки для распознавания, страница создания клеточных автоматов и работы с ними, страница комбинации клеточных автоматов в последовательности, страница запуска последовательности клеточных автоматов, страница обучения и классификации. Каждая страница – сущность, она реализует общий интерфейс, как показано на рис. П.2.

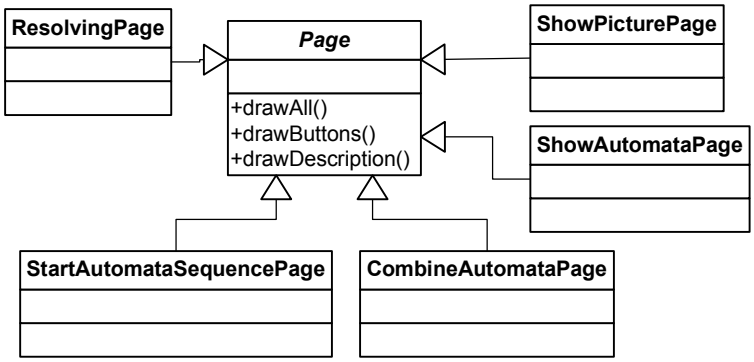


Рис. П.2. Диаграмма классов блока реализации страниц пользовательского интерфейса

В таблице П2 представлено описание классов блока реализации страниц пользовательского интерфейса.

Таблица П2. Описание классов блока реализации страниц пользовательского интерфейса

Класс/поле/метод	**Описание**
`Page` – интерфейс преобразования объектов в *XML* формат	
`Page previous`	Предыдущая страница интерфейса
`Page next`	Следующая страница интерфейса
`void drawAll()`	Обрисовка всех элементов страницы
`drawButtons()`	Обрисовка элемента кнопок страницы
`drawDescription()`	Обрисовка элемента описания страницы
`draw ()`	Обрисовка специфических элементов страницы. Метод реализован в потомках: `ShowPicturePage`, `ShowAutomataPage`, `ShowAutomataSequencePage`, `CombineAutomataPage`, `ResolvingPage`

П.3. Языковой блок

Моделирующая программа предусматривает различные языки интерфейса. Для добавления дополнительного поддерживаемого языка достаточно реализовать интерфейс локализации. Диаграмма классов блока показана на рис. П.3.

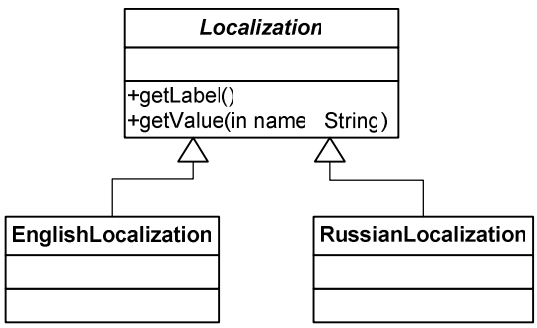


Рис. П.3. Диаграмма классов языкового блока

В табл. П3 представлено описание классов языкового блока.

Таблица П3. Описание классов языкового блока

Класс/поле/метод	**Описание**
`Localization` – абстрактный класс описания локализации программы	
`String getValue(String name)`	Возвращает значение по имени. Метод определен в потомках класса
`String getLabel()`	Возвращает уникальный индекс локализации
`RussianLocalization` – класс описания русской локализации программы	
`EnglishLocalization` – класс описания английской локализации программы	