

Иркутский государственный технический университет

На правах рукописи
УДК 519.6

КАТАШЕВЦЕВ МИХАИЛ ДМИТРИЕВИЧ

**Математическое моделирование контурных изображений и
вычислительная сложность их анализа**

Специальность 05.13.18 —

«Математическое моделирование, численные методы и комплексы программ»

Диссертация на соискание учёной степени
кандидата физико-математических наук

Научный руководитель:
д. ф-м. н., профессор
Мартынов В.И.

Иркутск – 2014

Содержание

Введение	5
1 Обзор	11
1.1	Теория распознавания образа	11
1.2	Распознавание текста	12
1.3	Персептрон	14
1.4	Алгоритмы обучения	17
1.4.1	Обучение с учителем	18
1.4.2	Обучение без учителя	19
1.4.3	Метод обратного распространения ошибки	19
1.5	Поисковые системы	20
1.5.1	Big Table	21
2 Распознавание изображений	23
2.1	Базовые понятия	23
2.2	Модель контурного изображение	27
2.3	Преобразование растрового изображения	29
2.3.1	Волновая скелетизация	30
2.3.2	Построение модели \mathfrak{M} для графа G	33
2.4	Постановка задачи анализа плоских контурных изображений	37
2.5	Оценка сложности анализа изображений	40
2.6	Оценка сложности анализа изображений с метрикой	47
2.7	Масштабные ряды	50
2.7.1	Результаты	54
3 Приложения	59
3.1	Распознавание символов	59

3.1.1	Конвертор растровых изображений	60
3.1.2	Браузер для БД	61
3.1.3	Интерпретатор	62
3.2	Оценка устойчивости битумных эмульсий	63
3.2.1	Введение	63
3.2.2	Анализ эмульсий	65
3.2.3	Оценка качества анализа	67
3.2.4	Определение среднего размера и дисперсии частиц битумной эмульсии на модифицированном битуме	69
3.2.5	Результаты	70
3.3	Автоматизация составления ПОДД	70
3.3.1	Представление дороги	72
3.3.2	Представление правил	73
3.3.3	Автоматизация	75
Заключение	79
Список рисунков	80
Список таблиц	81
Литература	82
А Структура БД	85

Введение

Разработка эффективных вычислительных математических методов решения информационных задач является важнейшим направлением развития современных информационных технологий (и в целом, прогресса, так как невозможно представить современную науку и технику без использования компьютеров).

В идеале, эти методы должны обеспечивать скорость решения ряда важных информационных задач вне зависимости от объема данных. И действительно, есть ряд важных информационных задач, где это возможно.

Наглядным (и очень важным) примером этого являются реляционные базы данных (БД), где вычислимость запросов определенных типов не зависит от объема данных, а только линейно от сложности проекта самой БД.

Практическим подтверждением этого для обывателя является скорость работы банковских систем, использующих сетевые реляционные БД, с их мировыми сетями терминалов и банкоматов, где можно проводить операции с вкладами и денежными средствами в любой точке мира за считанные секунды.

Отметим, что весьма близко к этому классу примыкают задачи поиска данных по ключевым словам в Интернет - пространстве (полно текстовые поисковые системы «Google» [1], «Яндекс» и др.), где скорость поиска не замедляется из-за экспоненциального роста информации в глобальной сети.

Другим важным классом информационных задач являются вопросы распознавания образцов (образов), где при организации данных близкой к таблицам реляционных БД также могут быть получены результаты независимости скорости распознавания образцов от их количества [2], вернее, верхней границы сложности распознавания одного образца с добавкой только количества образцов.

Данное направление исследований (разработка эффективных вычислительных математических методов решения информационных задач) весьма актуаль-

но для современной математики, науки в целом и техники, как в теоретическом, так и практическом плане, где работают крупные транснациональные компьютерные корпорации, реализуются технологии «Big Table», «Big Data» и предполагается получение прорывных результатов в робототехнике, молекулярной биологии, системах искусственного интеллекта и других важных областях, имеющих определяющее значение для прогресса современного общества [1, 3].

Целью данной работы является построение математических моделей контурных изображений, которые, хотя и имеют более сложную организацию данных, чем таблицы реляционных БД, но позволяют получить почти аналогичные результаты по вычислительной сложности анализа контурных изображений, включая проверку изоморфных вложений образцов в анализируемое изображение.

Разработка новых математических методов моделирования объектов и явлений, в диссертации представлена результатами по построению математических моделей контурных изображений.

Разработка, обоснование и тестирование эффективных вычислительных методов с применением современных компьютерных технологий в диссертации представлена результатами по эффективной вычислимости анализа, полученных математических моделей контурных изображений, включая проверку изоморфной вложимости совокупности контурных изображений (образцы) в исследуемое контурное изображение.

Реализация эффективных численных методов и алгоритмов в виде комплексов проблемно-ориентированных программ для проведения вычислительного эксперимента в диссертации представлена результатами по применению комплексов программ для решения конкретных прикладных задач.

Для достижения поставленной цели необходимо было решить следующие задачи:

1. Разработать преобразование растра контурного изображения в нагруженный граф специального вида.
2. Разработать преобразование нагруженных графов специального вида в математические модели, представленные многоосновными алгебраическими системами, где контурные изображения сведены к ориентированным ду-

гам, связям дуг и их численным характеристикам в градусном измерении и относительных размеров длины дуг.

3. Исследовать алгоритмическую сложность анализа контурных изображений, включая проверку изоморфных вложений образцов в анализируемое изображение, где контурные изображения сведены к ориентированным дугам, связям дуг и их численным характеристикам в градусном измерении и относительных размеров длины дуг.
4. Разработать масштабные ряды контурных изображений и процедуры сжатия на основе относительных размеров дуг.
5. Исследовать возможность использования изоморфного вложения сжатого образца в сжатое изображение для уменьшения алгоритмической сложности построения изоморфного вложения исходного образца в исходное изображение.
6. Исследовать возможность использования полученных математических методов, математических моделей представления данных, алгоритмов и комплексов программ для решения прикладных задач:
 - (а) распознавания символов;
 - (б) оценки устойчивости битумных эмульсий;
 - (с) автоматизации составления проектов организации дорожного движения (ПОДД).

Основные положения, выносимые на защиту:

1. Сходящийся алгоритм волновой скелетизации, обеспечивающий преобразование растра контурного изображения в нагруженный граф специального вида (Утверждение 2.3.1.).
2. Оценка нижней границы алгоритмической сложности анализа контурных изображений, включая проверку изоморфных вложений образцов в анализируемое изображение, где контурные изображения сведены к ориентированным дугам, связям дуг и их численным характеристикам в градусном измерении и относительным размерам длины дуг (Теорема 2.5.1.).

3. Комплексы программ решения прикладных задач:

- (а) распознавания символов;
- (б) оценки устойчивости битумных эмульсий;
- (с) автоматизации составления проектов организации дорожного движения (ПОДД).

Научная новизна:

1. Впервые построены математические модели контурных изображений, представленных ориентированными дугами, связям дуг и их численными характеристиками в градусном измерении, а также относительными размерами длин дуг.
2. Впервые показано, что математические модели контурных изображений, хотя и имеют более сложную организацию данных, чем таблицы реляционных БД, но позволяют получить почти аналогичные результаты по алгоритмической сложности анализа контурных изображений, включая проверку изоморфных вложений образцов в анализируемое изображение.

Научная и практическая значимость определяется:

1. актуальностью направления исследований, которое обеспечивает применение информационных технологий буквально во всех сферах современной деятельности общества;
2. применением разработанных программных комплексов для решения прикладных задач, что подтверждено официальными справками о применении результатов диссертации. Степень достоверности полученных результатов обеспечивается строгими математическими формулировками определений, а также строгими математическими доказательствами полученных утверждений, лемм и теорем.

Результаты находятся в соответствии с результатами, полученными другими авторами: А.И.Мальцевым [4], Ю.Л.Ершовым [4], С.В.Яблонским [5], А.И.Кокориным [6], А.В.Манциводой [7], Коддом [8, 9], Д.Кнутом [10], В.И.Мартыновым [2], Д.В.Пахомовым [11], В.В.Архиповым [12] и др [13–16].

Основные результаты работы докладывались на:

1. ежегодных научно-теоретических конференциях аспирантов и студентов: Иркутский гос. университет, ИМЭИ, 2010-13 гг;
2. 3-ей Российской школе – семинаре «Синтаксис и семантика логических систем». Иркутск, 2010;
3. 4-ой Международной конференции «Математика, ее приложения и математическое образование (МПМО'11), Улан-Удэ, 2011;
4. семинарах кафедр ИГУ, ИрГТУ, ВСГАО, 2010-14гг;

Автором получены самостоятельно результаты основных положений 1, 2, выносимых на защиту. Результаты основных положений 3,4 выносимых на защиту, получены в нераздельном соавторстве с В.И. Мартыновым, которому принадлежит начальное определение масштабных рядов контурных изображений и предложение их использования для уменьшения вычислительной сложности анализа контурных изображений.

Основные результаты по теме диссертации изложены в 10 печатных изданиях [?, 11, 12, 17–23], 7 из которых изданы в журналах, рекомендованных ВАК [?, 12, 18–20, 22, 23]. Диссертация состоит из введения, четырех глав, заключения и двух приложений. Полный объем диссертации составляет 90 страниц с 19 рисунками и 6 таблицами. Список литературы содержит XXX наименований.

Глава 1

Обзор

В данной работе анализ рассматривается в контексте задачи классификации. Рассмотрим наиболее популярные на момент публикации данной работы методы анализа образов.

1.1 Теория распознавания образа

Теория распознавания образа — раздел информатики и смежных дисциплин, развивающий основы и методы классификации и идентификации предметов, явлений, процессов, сигналов, ситуаций и т. п. объектов, которые характеризуются конечным набором некоторых свойств и признаков. Такие задачи решаются довольно часто, например, при переходе или проезде улицы по сигналам светофора. Распознавание цвета загоревшейся лампы светофора и знание правил дорожного движения позволяет принять правильное решение о том, можно или нельзя переходить улицу.

Необходимость в таком распознавании возникает в самых разных областях — от военного дела и систем безопасности до оцифровки аналоговых сигналов.

Проблема распознавания образа приобрела выдающееся значение в условиях информационных перегрузок, когда человек не справляется с линейно-последовательным пониманием поступающих к нему сообщений и в результате его голова переключается на режим одновременности восприятия и мышления, которому такое распознавание свойственно.

Неслучайно, таким образом, проблема распознавания образа оказалась в поле междисциплинарных исследований - в том числе в связи с работой по созданию искусственного интеллекта, а создание технических систем распознавания образа привлекает к себе всё большее внимание.

Распознавание образов — это отнесение исходных данных к определенному классу с помощью выделения существенных признаков, характеризующих эти данные, из общей массы несущественных данных.

Классическая постановка задачи распознавания образов: близка к постановке задачи классификации и предполагает наличие некоторого базового набора заранее известных классов (набор может расширяться), и образа (не обязательно визуального) которые надо классифицировать.

Наиболее часто в задачах распознавания визуальных образов рассматриваются монохромные изображения, что дает возможность рассматривать изображение как функцию на плоскости.

Множество же всех возможных функций $f(x, y)$ на плоскости T — есть модель множества всех изображений X . Вводя понятие сходства между образами можно поставить задачу распознавания. Конкретный вид такой постановки сильно зависит от последующих этапов при распознавании в соответствии с тем или иным подходом.

1.2 Распознавание текста

Оптическое распознавание текста является исследуемой проблемой в областях распознавания образов, искусственного интеллекта и компьютерного зрения.

Системы оптического распознавания текста требуют калибровки для работы с конкретным шрифтом; в ранних версиях для программирования было необходимо изображение каждого символа, программа одновременно могла работать только с одним шрифтом. В настоящее время больше всего распространены так называемые «интеллектуальные» системы, с высокой степенью точности распознающие большинство шрифтов. Некоторые системы оптического распознавания текста способны восстанавливать исходное форматирование текста, включая изображения, колонки и другие нетекстовые компоненты.

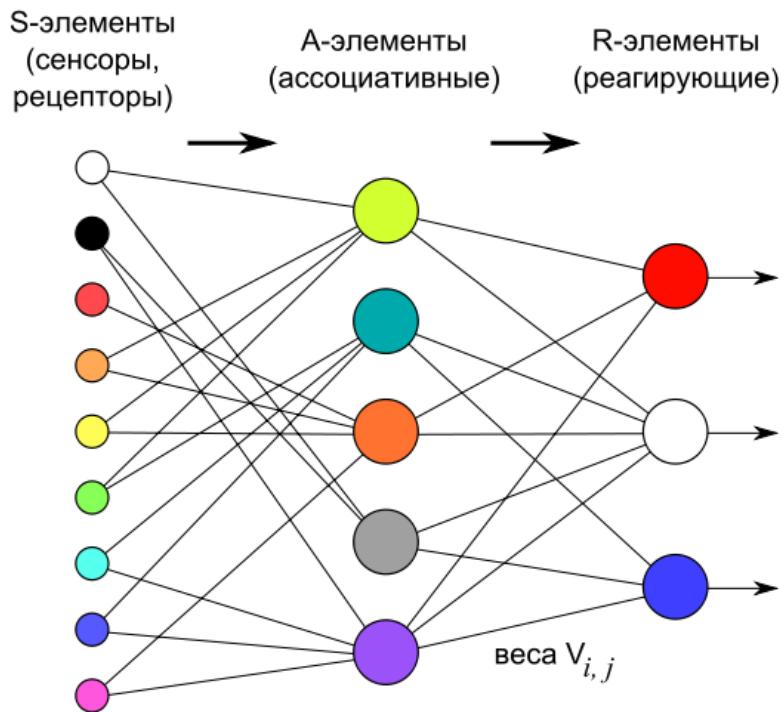
Точное распознавание латинских символов в печатном тексте в настоящее время возможно только если доступны чёткие изображения, такие как сканированные печатные документы. Точность при такой постановке задачи превышает 99%, абсолютная точность может быть достигнута только путем последующего редактирования человеком. Проблемы распознавания рукописного «печатного» и стандартного рукописного текста, а также печатных текстов других форматов (особенно с очень большим числом символов) в настоящее время являются предметом активных исследований.

Точность работы методов может быть измерена несколькими способами и поэтому может сильно варьироваться. К примеру, если встречается специализированное слово, не используемое для соответствующего программного обеспечения, при поиске несуществующих слов, ошибка может увеличиться.

Распознавание символов он-лайн иногда путают с оптическим распознаванием символов. Последний — это офф-лайн метод, работающий со статической формой представления текста, в то время как он-лайн распознавание символов учитывает движения во время письма. Например, в он-лайн распознавании, используемом PenPoint OS или планшетный ПК, можно определить, с какой стороны пишется строка: справа налево или слева направо.

Он-лайн системы для распознавания рукописного текста «на лету» в последнее время стали широко известны в качестве коммерческих продуктов. Алгоритмы таких устройств используют тот факт, что порядок, скорость и направление отдельных участков линий ввода известны. Кроме того, пользователь научится использовать только конкретные формы письма. Эти методы не могут быть использованы в программном обеспечении, которое использует сканированные бумажные документы, поэтому проблема распознавания рукописного «печатного» текста по-прежнему остается открытой. На изображениях с рукописным «печатным» текстом без артефактов может быть достигнута точность в 80% — 90%, но с такой точностью изображение будет преобразовано с десятками ошибок на странице. Такая технология может быть полезна лишь в очень ограниченном числе приложений.

Ещё одной широко исследуемой проблемой является распознавание рукописного текста. На данный момент достигнутая точность даже ниже, чем для рукописного «печатного» текста. Более высокие показатели могут быть достигнуты



только с использованием контекстной и грамматической информации. Например, в процессе распознания искать целые слова в словаре легче, чем пытаться проанализировать отдельные символы из текста. Знание грамматики языка может также помочь определить, является ли слово глаголом или существительным. Формы отдельных рукописных символов иногда могут не содержать достаточно информации, чтобы точно (более 98%) распознать весь рукописный текст.

Для решения более сложных проблем в сфере распознавания используются как правило интеллектуальные системы распознавания, такие как искусственные нейронные сети.

1.3 Персепtron

Ф. Розенблатт вводя понятие о модели мозга, задача которой состоит в том, чтобы показать, как в некоторой физической системе, структура и функциональные свойства которой известны, могут возникать психологические явления — описал простейшие эксперименты по различению. Данные эксперименты целиком относятся к методам распознавания образов, но отличаются тем, что алгоритм решения не детерминированный.

Простейший эксперимент, на основе которого можно получить психологически значимую информацию о некоторой системе, сводится к тому, что модели предъявляются два различных стимула и требуется, чтобы она реагировала на них различным образом. Целью такого эксперимента может быть исследование возможности их спонтанного различения системой при отсутствии вмешательства со стороны экспериментатора, или, наоборот, изучение принудительного различения, при котором экспериментатор стремится обучить систему проводить требуемую классификацию.

В опыте с обучением персептрону обычно предъявляется некоторая последовательность образов, в которую входят представители каждого из классов, подлежащих различению. В соответствии с некоторым правилом модификации памяти правильный выбор реакции подкрепляется. Затем персептрону предъявляется контрольный стимул и определяется вероятность получения правильной реакции для стимулов данного класса. В зависимости от того, совпадает или не совпадает выбранный контрольный стимул с одним из образов, которые использовались в обучающей последовательности, получают различные результаты:

Если контрольный стимул не совпадает ни с одним из обучающих стимулов, то эксперимент связан не только с чистым различием, но включает в себя и элементы обобщения. Если контрольный стимул возбуждает некоторый набор сенсорных элементов, совершенно отличных от тех элементов, которые активизировались при воздействии ранее предъявленных стимулов того же класса, то эксперимент является исследованием чистого обобщения.

Персептроны не обладают способностью к чистому обобщению, но они вполне удовлетворительно функционируют в экспериментах по различию, особенно если контрольный стимул достаточно близко совпадает с одним из образов, относительно которых персепtron уже накопил определенный опыт.

Элементарный перцепtron состоит из элементов 3-х типов: *S*-элементов, *A*-элементов и одного *R*-элемента. *S*-элементы — это слой сенсоров, или рецепторов. В физическом воплощении они соответствуют, например, светочувствительным клеткам сетчатки глаза или фоторезисторам матрицы камеры. Каждый рецептор может находиться в одном из двух состояний — покоя или возбуждения, и только в последнем случае он передаёт единичный сигнал в следующий слой, ассоциативным элементам.

A-элементы называются ассоциативными, потому что каждому такому элементу, как правило, соответствует целый набор (ассоциация) *S*-элементов. *A*-элемент активизируется, как только количество сигналов от *S*-элементов на его входе превысило некоторую величину θ . Таким образом, если набор соответствующих *S*-элементов располагается на сенсорном поле в форме буквы «Д», *A*-элемент активизируется, если достаточное количество рецепторов сообщило о появлении «белого пятна света» в их окрестности, то есть *A*-элемент будет как бы ассоциирован с наличием/отсутствием буквы «Д» в некоторой области.

Сигналы от возбудившихся *A*-элементов, в свою очередь, передаются в сумматор *R*, причём сигнал от *i*-го ассоциативного элемента передаётся с коэффициентом w_i . Этот коэффициент называется весом *A – R* связи.

Так же как и *A*-элементы, *R*-элемент подсчитывает сумму значений входных сигналов, помноженных на веса (линейную форму). *R*-элемент, а вместе с ним и элементарный перцептрон, выдаёт «1», если линейная форма превышает порог θ , иначе на выходе будет «-1». Математически, функцию, реализуемую *R*-элементом, можно записать так:

$$f(x) = \text{sign}\left(\sum_{i=1}^n w_i x_i - \theta\right)$$

Обучение элементарного перцептрана состоит в изменении весовых коэффициентов w_i связей *A – R*. Веса связей *S – A* (которые могут принимать значения $\{-1; \theta; +1\}$) и значения порогов *A*-элементов выбираются случайным образом в самом начале и затем не изменяются. (Описание алгоритма см. ниже.)

После обучения перцептрон готов работать в режиме распознавания или обобщения. В этом режиме перцептруну предъявляются ранее неизвестные ему объекты, и перцептрон должен установить, к какому классу они принадлежат. Работа перцептрана состоит в следующем: при предъявлении объекта, возбудившиеся *A*-элементы передают сигнал *R*-элементу, равный сумме соответствующих коэффициентов w_i . Если эта сумма положительна, то принимается решение, что данный объект принадлежит к первому классу, а если она отрицательна — то ко второму.

1.4 Алгоритмы обучения

Важным свойством любой нейронной сети является способность к обучению. Процесс обучения является процедурой настройки весов и порогов с целью уменьшения разности между желаемыми (целевыми) и получаемыми векторами на выходе. В своей книге Розенблatt пытался классифицировать различные алгоритмы обучения перцептрона, называя их системами подкрепления.

Система подкрепления - это любой набор правил, на основании которых можно изменять с течением времени матрицу взаимодействия (или состояние памяти) перцептрона.

1.4.1 Обучение с учителем

Классический метод обучения перцептрона — это метод коррекции ошибки. Он представляет собой такой вид обучения с учителем, при котором вес связи не изменяется до тех пор, пока текущая реакция перцептрона остается правильной. При появлении неправильной реакции вес изменяется на единицу, а знак (+/-) определяется противоположным от знака ошибки.

Допустим, мы хотим обучить перцептрон разделять два класса объектов так, чтобы при предъявлении объектов первого класса выход перцептрона был положителен (+1), а при предъявлении объектов второго класса — отрицательным (-1). Для этого выполним следующий алгоритм:

Случайным образом выбираем пороги для A -элементов и устанавливаем связи $S - A$ (далее они изменяться не будут). Начальные коэффициенты w_i полагаем равными нулю. Предъявляем обучающую выборку: объекты (например, круги либо квадраты) с указанием класса, к которым они принадлежат.

Показываем перцептрону объект первого класса. При этом некоторые A -элементы возбуждаются. Коэффициенты w_i , соответствующие этим возбуждённым элементам, увеличиваем на 1.

Предъявляем объект второго класса и коэффициенты w_i тех A -элементов, которые возбуждаются при этом показе, уменьшаем на 1.

Обе части шага 3 выполним для всей обучающей выборки. В результате обучения сформируются значения весов связей w_i .

Теорема сходимости перцептрона, описанная и доказанная Ф. Розенблаттом (с участием Блока, Джозефа, Кестена и других исследователей, работавших вместе с ним), показывает, что элементарный перцептрон, обучаемый по такому алгоритму, независимо от начального состояния весовых коэффициентов и последовательности появления стимулов всегда приведет к достижению решения за конечный промежуток времени.

1.4.2 Обучение без учителя

Кроме классического метода обучения перцептрона Розенблatt также ввёл понятие об обучении без учителя, предложив следующий способ обучения:

Альфа-система подкрепления - это система подкрепления, при которой веса всех активных связей c_{ij} , которые ведут к элементу u_j , изменяются на одинаковую величину r , а веса неактивных связей за это время не изменяются.

Затем, с разработкой понятия многослойного перцептрона, альфа-система была модифицирована и её стали называть дельта-правило. Модификация была проведена с целью сделать функцию обучения дифференцируемой (например, сигмоидной), что в свою очередь нужно для применения метода градиентного спуска, благодаря которому возможно обучение более одного слоя.

1.4.3 Метод обратного распространения ошибки

Для обучения многослойных сетей рядом учёных, в том числе Д. Румельхартом, был предложен градиентный алгоритм обучения с учителем, проводящий сигнал ошибки, вычисленный выходами перцептрона, к его входам, слой за слоем. Сейчас это самый популярный метод обучения многослойных перцептронов. Его преимущество в том, что он может обучить все слои нейронной сети, и его легко просчитать локально. Однако этот метод является очень долгим, к тому же, для его применения нужно, чтобы передаточная функция нейронов была дифференцируемой. При этом в перцептронах пришлось отказаться от бинарного сигнала, и пользоваться на входе непрерывными значениями.

1.5 Поисковые системы

Процессы компьютеризации деятельности государственных учреждений, предприятий, быстрое развитие глобальной сети (Internet) привели к накоплению большого объема неструктурированной текстовой информации. Возникла потребность в программном обеспечении, реализующем эффективный поиск информации.

Все это привело к созданию полнотекстовых информационно-поисковых систем. Полнотекстовые Поисковые Системы (ПС) строятся на основе информационно-поисковых языков дескрипторного типа. Информационно-технологическая структура полнотекстовых ИС включает:

1. хранилище документов;
2. глобальный словарь системы;
3. инвертированный индекс документов;
4. интерфейс ввода документов в систему;
5. механизм индексирования;
6. интерфейс запросов пользователя;
7. механизм поиска документов;
8. механизм извлечения найденных документов.

Приведем краткий перечень типов наиболее широко используемых полнотекстовых ПС.

1. Справочно-информационные системы общего назначения, ориентированные на доступ пользователей к нормативно-правовым документам. К этим системам относятся «Консультант Плюс», «Гарант», «Кодекс» и др.
2. Глобальные информационные службы (хост-системы), предоставляющие доступ удаленным пользователям к библиографической, полнотекстовой или другой информации. Крупнейшей в мире коммерческой службой, обеспечивающей доступ к юридической информации, является система LEXIS (США).

3. Системы информационной поддержки деятельности правотворческих органов. Спецификой таких систем является необходимость хранения и поиска многих версий и редакций нормативно - правовых документов, с учетом вносимых поправок и изменений.
4. Системы автоматизации делопроизводства судов, милиции и других правоохранительных органов.
5. Системы поиска информации в Интернете (Google, Yandex и др.).

Так же стоит отметить, что в настоящее время возможностями полнотекстового поиска обладают все наиболее популярные на сегодняшний день СУБД (SQL, MySQL, Postgres).

Наибольший интерес для нас будет представлять система поиска информации Google, основанная на технологии BigTable, которая более подробно будет рассмотрена в следующем разделе.

1.5.1 Big Table

Big Table – высокопроизводительная СУБД, построенная на основе Google File System (GFS), Chubby Lock Service и других программных продуктах Google. В настоящий момент не распространяется и не используется за пределами Google, хотя клон Bigtable применяется в Агентстве национальной безопасности США.

Технология Big Table является на настоящий момент наиболее мощной реализацией идеологии табличной организации данных и именно она обеспечивает корпорации Google лидирующие позиции на рынке оказания услуг по полнотекстовому поиску информации в глобальной сети, а также в сотрудничестве на коммерческой основе с различными государственными учреждениями США.

Отметим, что индексирование записей в СУБД Big Table не отличается от принятой в промышленных реляционных СУБД, если отбросить возможность масштабирования, т.е. размер индексов, и идеологически родственна представлению узлов в универсуме для изображений из раздела диссертации 2.5. Интерпретация.

Глава 2

Распознавание изображений

2.1 Базовые понятия

Определение 2.1.1. *Растровое изображение есть функция*

$$I_{rgb}(x, y) : N \times N \rightarrow ([0, 255], [0, 255], [0, 255])$$

Таким образом каждой точке (x, y) мы сопоставляем тройку (r, g, b) . Первый элемент тройки соответствует красной компоненте цвета в растровом изображении, второй – зеленой и третий – синей. Далее, для краткости, будем использовать следующую запись:

$$I_r(x, y) = I_{rgb}(x, y)_r \text{ – красная компонента}$$

$$I_g(x, y) = I_{rgb}(x, y)_g \text{ – зеленая компонента}$$

$$I_b(x, y) = I_{rgb}(x, y)_b \text{ – синяя компонента}$$

Определение 2.1.2. *Определим растровое изображение заданного в оттенках серого как функцию*

$$I_{grey}(x, y) : N \times N \rightarrow [0, 255]$$

Определение 2.1.3. *Введем оператор «обесцвечивания» D , который обеспечивает переход от цветного изображения I_{rgb} к изображению I_{grey} , заданному в оттенках серого:*

$$D(I_{rgb}) = I_{grey}$$

Существует несколько основных способов обесцвечивания изображения:

1. Красный канал $D_{red}(I_{rgb}) = I_r$
2. Зеленый канал $D_{green}(I_{rgb}) = I_g$
3. Синий канал $D_{blue}(I_{rgb}) = I_b$
4. Среднее значение (average): $D_{avg}(I_{rgb}) = \frac{I_r + I_g + I_b}{3}$
5. Лума (luma), учитывает особенности восприятия цвета человеком:

$$D_{luma}(I_{rgb}) = I_r \cdot 0.3 + I_g \cdot 0.59 + I_b \cdot 0.11$$

Значения коэффициентов, иногда, могут отличаться от приведенных выше, но их сумма всегда равна 1

6. Минимум $D_{min}(I_{rgb}) = \min(I_r, I_g, I_b)$
7. Максимум $D_{max}(I_{rgb}) = \max(I_r, I_g, I_b)$
8. Обесцвечивание (desaturation): $D_{desaturation}(I_{rgb}) = \frac{D_{min}(I_{rgb}) + D_{max}(I_{rgb})}{2}$

Замечание 2.1. Наилучший результат для средне-статистического изображения (с гистограммой близкой к нормальной) получается при использовании 5-го и последнего способов. Под наилучшим результатом понимается сохранение яркости (компоненты *value* в модели HSV) цветов исходного изображения.

Определение 2.1.4. Растровое монохромное изображение есть функция

$$I_m(x, y) : N \times N \rightarrow \{0, 1\}$$

Переход от изображения заданного в оттенках серого к монохромному изображению осуществляется через операцию отсечения. Операция отсечения реализуется через оператор отсечения T , для некоторого фиксированного $t \in [0, 255]$

Определение 2.1.5. Оператор отсечения T есть:

$$T(t, I_{grey}) = \begin{cases} 1 & , I_{grey} < t \\ 0 & , \text{иначе} \end{cases}$$

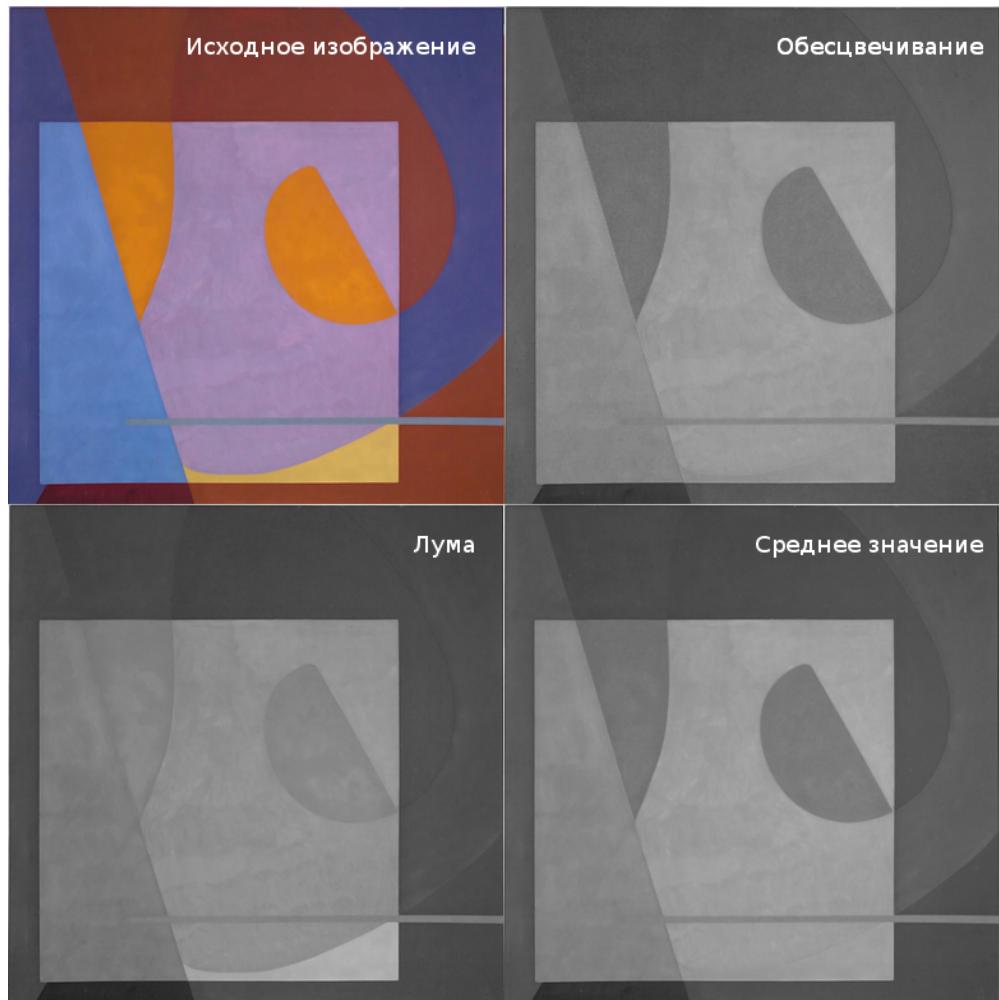


Рисунок 2.1: Сравнение методов обесцвечивания

Таким образом монохромное изображение есть $I_m = T(t, I_{grey})$

Замечание 2.2. Значение t определяется опытном путем и зависит от исходного изображения. Для рукописного текста написанного черной ручкой на офисной бумаге берутся значения близкие к 100 (чем меньше значение, тем темнее изображение).

Замечание 2.3. В данной работе не рассматриваются методы адаптивного отсечения, в силу их медленной производительности излишней в данном контексте точности.

Определение 2.1.6. Точки (x,y) для которых верно $I(x,y) = 1$ будем называть *заполненными*.

Определение 2.1.7. Точки (x,y) для которых верно $I(x,y) = 0$ будем называть *пустыми*.

Множество заполненных точек образует множество связных областей

$$\{K_1, K_2 \dots K_n\},$$

таких что:

1. $\forall(x_1, y_1) \forall(x_2, y_2) (|x_1 - x_2| > 1 \wedge |y_1 - y_2| > 1)$, где $(x_1, y_1) \in K_i, (x_2, y_2) \in K_j$ и $i \neq j$
2. $\forall(x_1, y_1) \exists(x_2, y_2) (|x_1 - x_2| \leq 1 \wedge |y_1 - y_2| \leq 1)$, где $(x_1, y_1), (x_2, y_2) \in K_i$ и $(x_1, y_1) \neq (x_2, y_2)$
3. $|K_i| \geq 2$

Определение 2.1.8. Всякую связную область K_i будем называть контуром

Определение 2.1.9. Растворное изображение I содержащие по крайней мере один контур будем называть растровым контурным изображением

Замечание 2.4. Не исключая общности, далее будут рассматриваться только растровые изображения содержащие один контур, а под растровым контурным изображением будет пониматься растровое изображение содержащие только один контур.

2.2 Модель контурного изображение

В качестве математической модели представления растрового контурного изображения будем использовать четырех-основную алгебраическую систему вида.

Определение 2.2.1. Контуровое изображение (далее, изображение) есть система вида

$$\mathfrak{M} = \langle A, R, V, M; Sector, Angle, Metric, Relation \rangle \quad (2.1)$$

где

A – множество всевозможных дуг,

R – множество связей дуг,

$V \subset Z$ – множество допустимых углов (например от 0 до 360 градусов),

$M \subset Z$ – множество относительных мер,

$Sector : A \rightarrow V$ – задает градусную меру дуги,

$Metric : A \rightarrow M$ – функция сопоставляющая каждой дуге ее относительную величину,

$Angle : R \rightarrow V$ – задает угол соединения двух дуг

$Relation : R \rightarrow A \times A$ – сопоставляет каждой связи дуги, те дуги, которые она соединяет.

Замечание 2.5. Важно отметить, что в этом представлении все множества являются конечными, и, если множества A и R являются фиктивными (чисто техническими элементами) данной модели, и определяются через функции, то множество $V = \{v_0, v_1, \dots, v_n\}$ – есть конечное множество чисел, с максимальным v_{max} и минимальным v_{min} элементами, разбитое шагом δ на $n + 1$ элементов, где

$$n = \frac{v_{max} - v_{min}}{\delta}$$

$$v_0 = v_{min}$$

$$v_i - v_{i-1} = \delta, \forall i = 1, n$$

$$v_n = v_{max}$$

Замечание 2.6. Как и множество V , множество M конечно. Однако выбор верхней границы для M не столь очевиден, так как не исключена возможность того, что разница в размерах между двумя дугами может быть весьма существенна (например в несколько миллионов раз). Однако, в рамках нашей области применения (распознавание символов), когда в качестве «эталонного наблюдателя» выступает человеческий глаз, разница что в 1000, что в 1000000 раз почти неразличима, и поэтому ею вполне можно пренебречь, выбрав в качестве максимального значения например 100000 процентов, а в качестве шага одну десятую процента. Таким образом всякая дуга может быть как больше так и меньше любой дуги не более чем в 1000 раз.

Для наших целей важно всегда работать только с конечными множествами, что достигается рассмотрением конечных множеств A , R , а также предположением о наличии минимального шага возрастания количественных характеристик дуг и связей дуг.

Таким образом контурное изображение будет представлять собой систему дуг и связей дуг. Где всякая дуга определяется через ее градусную меру и через относительную (в данном контуре) длину дуги. А всякая связь определяется через угол связи и пару дуг которые она связывает.

2.3 Преобразование растрового изображения

Переход от растрового контурного изображения к изображению состоит из двух этапов. Первый этап — волновая скелетизация. С помощью скелетизации на основе растрового изображение строится граф (скелет), который визуально адекватно соответствует исходному изображению.

Пусть I — растровое контурное изображение и K — есть его контур.

Определение 2.3.1. Точку $q(x_1, y_1)$ будем называть соседом точки $p(x, y)$ если $|x - x_1| \leq 1$ и $|y - y_1| \leq 1$ и $p \neq q$. Введем отношение соседства $N(p, q)$, которое истинно если p сосед q .

Очевидно что точка p не может иметь более 8 соседей. Обозначим через N_K^p множество всех соседей точки $p(x, y)$ лежащих контуре K :

$$N_K^p = \{q | q \in K \wedge N(p, q)\}$$

Согласно определению контура (2.1.8) очевидно, что K не имеет изолированных точек т.е.

$$\forall p \exists q : N(p, q)$$

$$p, q \in K$$

Замечание 2.7. Скелетом I будем называть граф $G(V, E)$, «интуитивно адекватно отражающий» исходное изображение.

Определение 2.3.2. Волной w будем называть конечное множество точек $\{p_j\}$.

Определение 2.3.3. Множество волн $\{w_1, w_2, \dots, w_n\}$ будем называть подволнами волны w если:

$$\bigcup_{i=1,n} w_i = w$$

$$w_i \cap w_j = \emptyset, i, j = \overline{1, n}, i \neq j;$$

и для любых двух точек $p \in w_i, g \in w_j$, где $i \neq j$ верно $\neg N(p, g)$.

Введем функцию вычисляющую центр масс точек волны

$$g(w) = \frac{\sum_{p \in w} p}{|w|}$$

2.3.1 Волновая скелетизация

Опишем алгоритм построения скелета растрового контурного изображения.

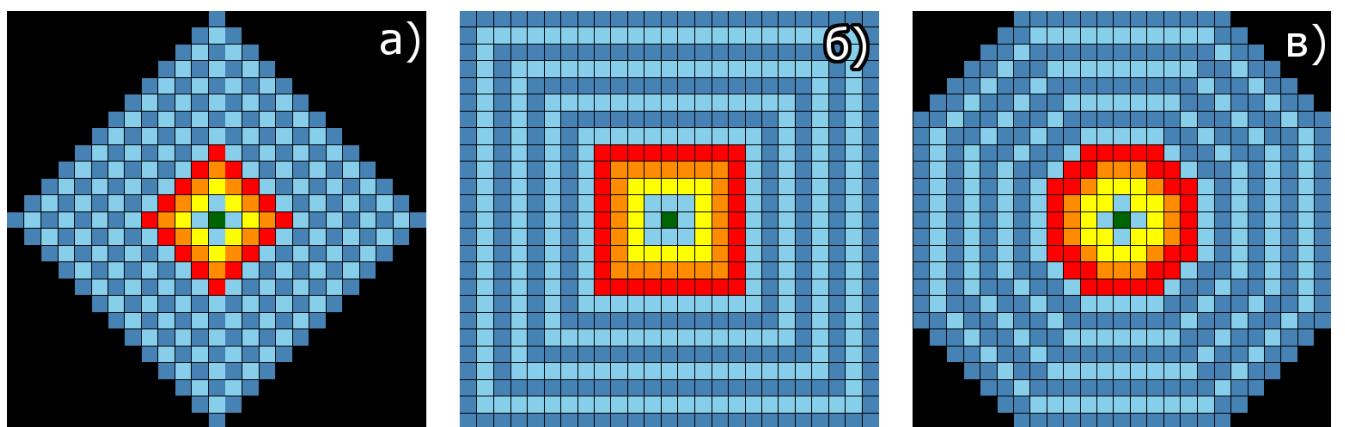


Рисунок 2.2: Распространение волны: а)-ромбовидная, б)-квадратная, в)-сферическая

Зададим начальные условия. В качестве начальной волны подойдет любая точка из F . Имеем следующую начальную конфигурацию:

$w_0^0 = \{p\}, p \in K$ – начальная волна,

$W_0 = \{w_0^0\}$ – множество волн,

$F_0 = K$ – состояние заполненной области,

$G_0(V_0, E_0), V_0 = \{p\}, E_0 = \emptyset$ - начальное состояние скелета.

Определим n -ый шаг итерации следующим образом. Для всякой i -ой волны $w_i^{k_i-1}$ из W_{n-1} (k_i - соответствует k_i -ой итерации w_i):

$$w_i^{k_i} = \bigcup_{p \in w_i^{k_i-1}} N_{F_{n-1}}^p \setminus \bigcup_{j < i} w_j^{k_j-1} \quad (2.2)$$

Если u_1, \dots, u_m есть подволны волны $w_i^{k_i}$, тогда

$$W_n^i = \{w_{l+1}, \dots, w_{l+m}\}$$

где

$$w_{l+j} = u_j, j = \overline{1, m}$$

$$l = |W_{n-1}| + \sum_{j < i} |W_n^j|.$$

Ребра в графе образуют вектора, связывающие центры масс полученных подволн с центром массы $w_i^{k_i-1}$

$$E_n^i = \{(v_i^{k_i-1}, v_{l+j}^{k_i})\}, j = \overline{1, m}$$

$$V_n^i = \{v_{l+j}^{k_i}\}, j = \overline{1, m}$$

$$v_i^j = g(w_i^j)$$

Если же волна $w_i^{k_i}$ не имеет разрывов и $w_i^{k_i} \neq \emptyset$, то

$$W_n^i = \{w_i^{k_i}\}$$

$$E_n^i = \{(v_i^{k_i-1}, v_i^{k_i})\}$$

$$V_n^i = \{v_i^{k_i}\}$$

Если $w_i^{k_i} = \emptyset$

$$W_n^i = \emptyset, E_n^i = \emptyset, V_n^i = \emptyset$$

Таким образом, при $s_n = |W_{n-1}|$:

$$W_n = \{W_n^i\}, i = \overline{1, s_n}$$

$$V_n = V_{n-1} \bigcup_i V_n^i, i = \overline{1, s_n}$$

$$E_n = E_{n-1} \bigcup_i E_n^i, i = \overline{1, s_n}$$

$$F_n = F_{n-1} \setminus P_{n-1},$$

$$P_{n-1} = \{p : p \in w, w \in W_{n-1}\}$$

Если $|F_n| = 0$ то алгоритм прекращает цикл итераций, а граф

$$G = (V_n, E_n)$$

является скелетом исходного изображения f .

Утверждение 2.3.1. Алгоритм волновая скелетизация остановится на всяком контуре мощности m

Доказательство. Пусть $m = 1$, тогда

$$|F_1| = |F_0 \setminus \{p : p \in w, w \in W_0\}| = |\{p\} \setminus \{p : p \in w_0\}| = |\{p\} \setminus \{p\}| = \emptyset$$

следовательно алгоритм прекращает свою работу а график $G = (V_1, E_1) = (\{p\}, \emptyset)$ является скелетом изображения.

Пусть $m > 2$. От противного: допустим, что существует такое l что для всякого $k < l, |F_k| < |F_{k-1}|$ и $|F_l| = |F_{l-1}|$, тогда $P_{n-1} = \emptyset$, отсюда следует, что $\{p : p \in w, w \in W_{n-1}\} = \emptyset$, а это возможно только в двух случаях:

1. если W_{n-1} пусто, тогда, в силу 2.2 и в силу отсутствия изолированных точек, $F_{n-2} = \emptyset$, получаем противоречие с условием остановки.
2. если $\forall w \in W : |w| = 0$, тогда, опять же в силу 2.2 и в силу отсутствия изолированных точек, получаем что $F_{n-2} = \emptyset$, снова получаем противоречие с условием остановки.

Следовательно такого l не существует, и алгоритм сходится для всякой непустой заполненной области без изолированных точек. \square

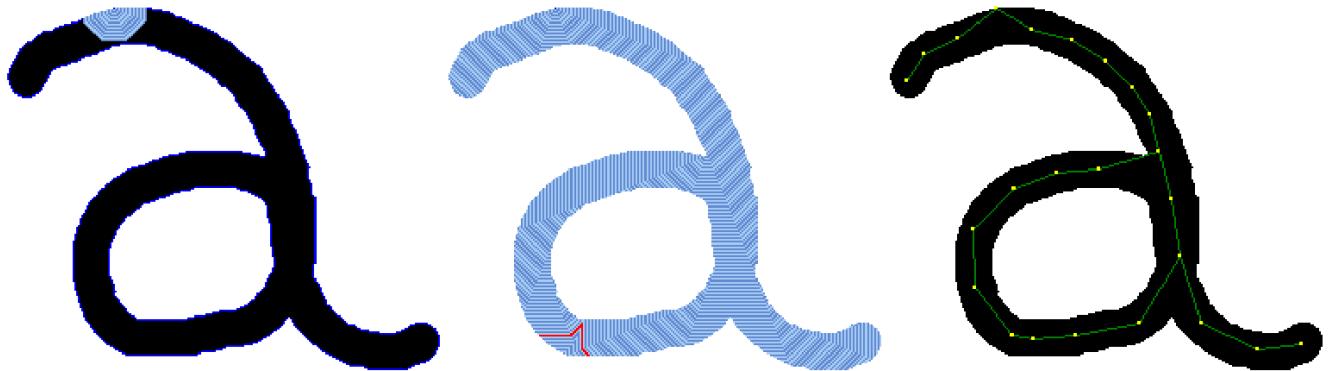


Рисунок 2.3: Распространение волны по области, на последнем изображении представлен полученный граф

2.3.2 Построение модели \mathcal{M} для графа G

Рассмотрим схему разбиения графа особыми точками

1. Для каждого простого пути выполняется:
 - (a) Разбиение пути по точкам смены направления обхода
 - (b) Для каждого разбиения выполняются
 - i. Разбиение спиралей. Чтобы определить закручен ли путь по спирали, надо проверить пересекает ли хорда путь. Если пересечение есть, то необходимо разбить путь точками пересечения
 - ii. Для каждого разбиения выполняется:
 - A. Разбиение по точкам перегиба. Точками перегиба считаются образующие две дуги отклонившиеся от угла идеального соединения. Угол γ идеального соединения двух дуг градусной меры α и β :

$$\gamma = 2\pi - (\alpha + \beta)$$

2. Результатом п.1 является множество подпутей, каждый из которых переводится в дугу. Градусная мера дуги вычисляется с использованием формулы

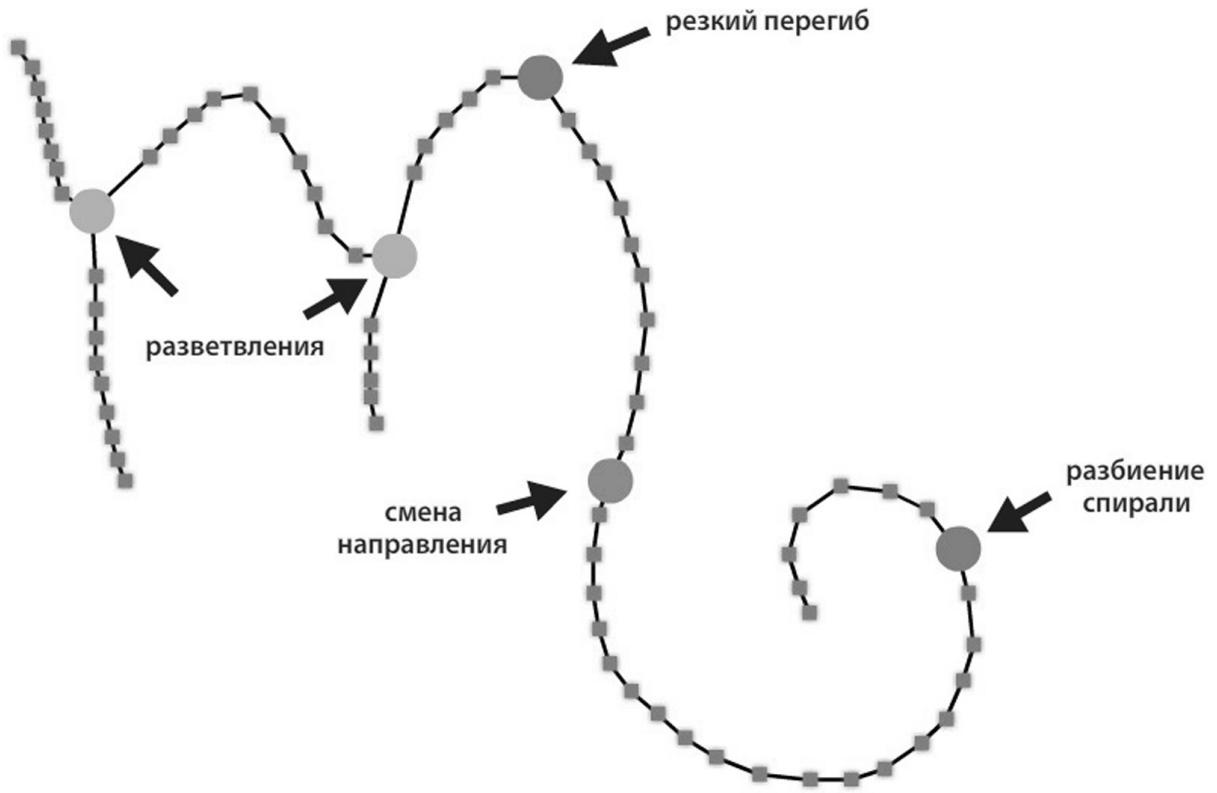


Рисунок 2.4: Схема разбиения графа «особыми» точками

Гюйгенса. Определив наиболее удаленную точку пути от хорды стягивающей путь, и вычислив её положение относительно хорды направленной от начала к концу пути, мы определяем направление обхода. Если точка слева, то обход ведется по часовой стрелке, если точка справа — обход ведется против часовой стрелки, если же точка лежит на прямой, то верны оба утверждения.

3. Расчет связей дуг. Связь между двумя дугами существует, если пути образующие дуги имели общие вершины. Угол соединения между дугами рассчитывается, как угол между стягивающими их хордами

Замечание 2.8. Стоит отметить что разбиение на дуги, как правило, выполняется на интерполированном графике в котором часть узлов удаленно в силу их избыточности.

Замечание 2.9. Хотя разбиение и может быть использовано как есть, на практике полезнее добавлять возможность вариации параметров, например, допускать возможность некоторого отклонения от угла идеального соединения или

для точек смены направления расширять область проверки на смену направления обхода.

Пусть $\{P_1, P_2, \dots, P_m\}$ – множество простых цепей графа $G = (V, E)$ таких что:

1. $P_i = v_1^i, v_2^i, \dots, v_{n_i}^i$
2. $\forall v \in P_i \forall u \in P_j (v \neq u)$, где $i \neq j$
3. $d(v_1^i) \neq 2$ и $d(v_{n_i}^i) \neq 2$
4. $\forall j \notin \{1, n_i\} [d(v_j^i) = 2]$
5. $\bigcup_i P_i = V$

Определение 2.3.4. Пусть $r(p, v)$ – есть расстояние от точки p до вектора v со знаком

Определение 2.3.5. Будем говорить что в узле v_i меняется направление обхода простого

$$\{v_1, \dots, v_{i-1}, v_i, v_{i+1}, \dots, v_n\}$$

в окрестности $\varepsilon \in \{2, 3, \dots\}$, если

$$Sign(\sum_{j=i-\varepsilon} r(v_j, h)) \neq Sign(\sum_{j=i+\varepsilon} r(v_j, h))$$

где $h = (v_{i-\varepsilon}, v_{i+\varepsilon})$

Если соотношение выполняется для $\varepsilon = 2$ будем просто говорить, что в точке v_i меняется направление обхода.

Пусть множество точек $\{v_{j_1^i}, \dots, v_{j_{k_i}^i}\}$ множество точек смены направления обхода цепи P_i . Определим множество индексов задающих разбиение пути $P_i = v_1, \dots, v_n$ по направлению обхода

$$I_{P_i}^{dev} = \{j_1^i, \dots, j_{k_i}^i\} \cup \{1, n\}$$

$$j_k^i < j_{k+1}^i$$

Тогда следующие семейство множеств узлов определяют разбиение пути P_i на подпути:

$$S_{P_i}^{dev} = \bigcup_{l \in I_{P_i}^{dev}} \{\{v_t \mid t \in \{j \mid i_l \leq j \leq i_{l+1}\}\}\}$$

Каждому множеству узлов можно однозначно сопоставить подпуть P_i

Определение 2.3.6. *Определим множество индексов узлов задающие разбиение подпути $P = v_1, \dots, v_n$ закрученного в спираль следующим образом*

$$I_P^{spir} = \left\{ \min_{|r(h, v_i)|} (i, i+1) \mid \text{Sign}(r(h, v_i)) \neq \text{Sign}(r(h, v_{i+1})) \wedge i = 1, n-1 \right\}$$

$$h = (v_1, v_n)$$

Определение 2.3.7. *Определим множество индексов узлов задающие разбиение подпути $P = v_1, \dots, v_n$ по резкости угла перегиба.*

Определение 2.3.8. *Разбиением пути P на характеристические подпути будем считать разбиение определенное следующим образом:*

$$S_{P_i} = \bigcup_{j \in I_P^{spir}, P \in S_{P_i}^{dev}} \{\{v_t \mid t \in \{j \mid i_l \leq j \leq i_{l+1}\}\}\}$$

Определение 2.3.9. *Пусть $P = v_1, \dots, v_n$ простой путь тогда:*

1. $f_{begin}(P) = v_1$ – начало пути
2. $f_{end}(P) = v_n$ – конец пути
3. $h_P = v_n - v_1$ – вектор хорды стягивающей путь

Определение 2.3.10. *Пусть задано разбиение графа $G = (V, E)$ на характеристические подпути $S = \{P_1, \dots, P_m\}$, где $P_i = v_1^i, \dots, v_{n_i}^i$, будем говорить что алгебраическая система*

$$\mathfrak{M} = \langle A, R, V, M; Sector, Angle, Metric, Relation \rangle$$

соответствует графу G если выполняются следующие условия:

1. $\text{Sector}(a_i) = 2l + \frac{1}{3}(2l - L)$, где $l = |v_1^i - v_m^i|$, $i_m = \frac{n_i}{2}$, а $L = |h_{P_i}|$

$$2. Metric(a_i) = \frac{a_i}{a_1}$$

3. $Relation(r_{ij}) = (a_i, a_j)$, где $i \neq j$ и неопределено в противном случае

$$4. Angle(r_{ij}) = \begin{cases} (-\hat{h_{P_i}}, -\hat{h_{P_j}}) & f_{begin}(P_i) = f_{end}(P_j) \\ (-\hat{h_{P_i}}, \hat{h_{P_j}}) & f_{begin}(P_i) = f_{begin}(P_j) \\ (\hat{h_{P_i}}, \hat{h_{P_j}}) & f_{end}(P_i) = f_{begin}(P_j) \\ (\hat{h_{P_i}}, -\hat{h_{P_j}}) & f_{end}(P_i) = f_{end}(P_j) \\ \text{неопределено, } & \text{если } r_{ij} - \text{неопределено} \end{cases}$$

2.4 Постановка задачи анализа плоских контурных изображений

Пусть изображение есть модель следующего вида:

$$\mathfrak{M} = \langle A, R, V; Sector, Angle, Relation \rangle \quad (2.3)$$

$$M_{ini} = \langle A_1, \dots, A_s; f_1, \dots, f_n; p_1, \dots, p_k \rangle \quad (2.4)$$

где A_i – основные множества, f_i – операции (функции) на основных множествах, p_i – предикаты (отношения) на основных множествах, в конечную (финальную) M_{fin} , удовлетворяющую ограничениям R_1, R_2, \dots, R_m .

Если искать аналоги, то последовательность таких преобразований можно считать допустимым (без оптимизации значений целевого функционала) управлением для задачи динамического программирования [?], где R_1, R_2, \dots, R_m фазовые ограничения.

При программной реализации многоосновная а.с. 2.4 становится реляционной БД, поиск последовательности преобразований для построения финальной а.с., удовлетворяющей ограничениям – комбинаторной задачей высокой сложности.

Для рассматриваемой здесь предметной области (анализ изображений) общая схема решения пока не может быть применена в полном объеме из-за начального этапа исследований (с точки зрения логико-эвристических методов) и

отсутствия конкретных постановок прикладных задач (ближайшие планы применения логико-эвристических методов обсуждаются в заключении).

Поэтому ограничимся исследованием сложности проверки выполнимости ограничений R_1, R_2, \dots, R_m на математических моделях вида 2.3 с позиции обеспечения независимости скорости проверки от числа ограничений.

Проверка выполнимости ограничений сводится к проверке вложимости обобщенных изображений (образцов) в анализируемое изображение 2.3.

Уточним формализацию описания плоских контурных изображений для данного варианта проверки выполнимости ограничений.

Составляющими элементами образцов и анализируемого изображения будут также дуги и связи дуг. Численными характеристиками которых будут количество минимальных шагов возрастания для дуги (связи дуг) – градусные меры дуг, имеющие минимальное и максимальное значения.

Таким образом, одноместная функция $Sector : Arc \rightarrow V$ преобразуется в одноместную же функцию $Sector : Arc \rightarrow V \times V$, соответственно, одноместная функция $Angle : Rel \rightarrow V$ преобразуется в одноместную же функцию $Angle : Rel \rightarrow V \times V$.

Замечание 2.10. В принципиальном плане ввод минимальных и максимальных значений ничего нового не дает (от минимального до максимального всего конечное множество значений), но позволяет более компактно задавать искомые образцы. Для упрощения технических деталей будем считать, что для анализируемых изображений функции $Sector$ и $Angle$ имеют одинаковое минимальное и максимальное значение

Пусть многоосновные а.с.

$$\begin{aligned} R_1 &= \langle Arc_1, Rel_1, V; Sector, Angle, R \rangle \\ R_2 &= \langle Arc_2, Rel_2, V; Sector, Angle, R \rangle \\ &\dots \\ R_m &= \langle Arc_m, Rel_m, V; Sector, Angle, R \rangle \end{aligned} \tag{2.5}$$

задают искомые образцы в анализируемом изображении 2.3.

Анализ изображения 2.3 состоит в поиске всех изоморфных вложений $\mu_{i,j}$ многоосновных а.с. R_1, R_2, \dots, R_m в многоосновную а.с. M 2.3, т.е. изоморфное

вложение $\mu_{i,j} : R_i \rightarrow M$ состоит из инъективных отображений

$$\mu_{i,j} : Arc_i \rightarrow Arc; \mu_{i,j} : Rel_i \rightarrow Rel \quad (2.6)$$

такие, что:

- a) если $\mu_{i,j}(Ar) = Arr$, где $Ar \in Arc_i$, $Arr \in Arc$, $Sector(Ar) = (v_1, v_2)$, $Sector(Arr) = (v_3, v_4)$, то $v_1 \leq v_3 \leq v_4 \leq v_2$;
- б) если $\mu_{i,j}(Re) = Ree$, где $Re \in Rel_i$, $Ree \in Rel$, $Angle(Re) = (v_1, v_2)$, $Angle(Ree) = (v_3, v_4)$, то $v_1 \leq v_3 \leq v_4 \leq v_2$;
- в) если $R(Re, Ar_1, Ar_2)$, где $Re \in Rel_i$, $Ar_1 \in Arc_i$, $Ar_2 \in Arc_i$, то $R(\mu_{i,j}(Re), \mu_{i,j}(Ar_1), \mu_{i,j}(Ar_2))$.

2.5 Оценка сложности анализа изображений

Для облегчения понимания идеи доказательства основного результата, рассмотрим доказательство теоремы. Основой его является представление декартона произведения конечных множеств.

$$A_1 \times A_2 \times \dots \times A_n$$

в древовидной форме Tree [?].

Определим более точно конечные множества

$$\begin{aligned} A_1 &= \{a_{1,1}, a_{1,2}, \dots, a_{1,m_1}\}; \\ A_2 &= \{a_{2,1}, a_{2,2}, \dots, a_{2,m_2}\}; \\ &\dots \\ A_n &= \{a_{n,1}, a_{n,2}, \dots, a_{n,m_n}\}; \end{aligned} \quad (2.7)$$

Понятно, что таблица 1 является универсумом для любых таблиц реляционной БД с доменами A_1, A_2, \dots, A_n . Т.е представление отношения H в таблице состоит в пометке вершин n -го этажа, если путь от корня дерева до этой вершины n -го этажа, дает кортеж из отношения H .

Проверка принадлежности кортежа (a_1, a_2, \dots, a_n) , где $a_1 \in A_1, a_2 \in A_2, \dots, a_n \in A_n$, отношению H производится за n шагов в таблице 1(этот процесс в дальнейшем будем называть интерпретацией кортежа (a_1, a_2, \dots, a_n) на дереве $Tree$). Действительно, a_1 позиционируется на 1-ом этаже за 1 шаг, a_2 позиционируется на 2-ом этаже за 1 шаг и так далее, a_n позиционируется на n -ом этаже за 1 шаг, где и определяется принадлежность кортежа (a_1, a_2, \dots, a_n) отношению H .

Таким образом, проверка осуществляется за n шагов. Для доказательства теоремы достаточно пометить вершины n -го этажа на принадлежность отношениям H_1, H_2, \dots, H_k . Тогда в результате интерпретации кортежа (a_1, a_2, \dots, a_n) за n шагов на дереве $Tree$ получим вершину n -го этажа пометки которой покажут принадлежность (или не принадлежность) отношениям H_1, H_2, \dots, H_k .

Оценка $O(n+k)$, а не $O(n)$, получается из-за необходимости пройти по списку отметок n -го этажа, что и дает добавку $O(k)$.

Замечание 2.11. Результат теоремы ?? типичный, так называемый обмен памяти на эффективность [?]. Конечно, задание декартово произведения деревом увеличивает необходимый объем памяти, но скорость выполнения операций предельно ускоряется. Следует отметить также, что на практике универсум (схема 1) не строится, а строится только его часть, состоящая из кортежей отношений H_1, H_2, \dots, H_k . Вообще говоря, это замедляет скорость интерпретации, но незначительно не более, чем на $\ln(m)$, где $m = \max\{m_1, m_2, \dots, m_n\}$ (это связано с необходимостью перебора узлов "частичного" дерева, что, в силу упорядоченности, можно реализовать с помощью бинарного поиска).

Прежде, чем перейти к изложению основного результата, определим универсум для изображений, имеющих не более n дуг, и k вариантов дуг и связей дуг, т.е. множество V имеет k элементов, а минимальный сектор дуги или угол пересечения дуг равен $(360/k)$ градусов.

Пусть $Arc_1, Arc_2, \dots, Arc_n$ – множества дуг всех характеристик (образцов), т.е.

$$\begin{aligned} Arc_1 &= \{ar_{1,1}, ar_{1,2}, \dots, ar_{1,m_1}\}; \\ Arc_2 &= \{ar_{2,1}, ar_{2,2}, \dots, ar_{2,m_2}\}; \\ &\dots \\ Arc_n &= \{ar_{n,1}, ar_{n,2}, \dots, ar_{n,m_n}\}; \end{aligned} \tag{2.8}$$

Далее пусть $Rel_1, Rel_2, \dots, Rel_{n-1}$ – множества связей дуг всех характеристик, т.е.

$$\begin{aligned}
 Rel_1 &= \{re_{1,1}, re_{1,2}, \dots, re_{1,k_1}\}; \\
 Rel_2 &= \{re_{2,1}, re_{2,2}, \dots, re_{2,k_2}\}; \\
 &\dots \\
 Rel_{n-1} &= \{re_{n-1,1}, re_{n-1,2}, \dots, re_{n-1,k_{n-1}}\}; \\
 Angle(re_{i,j}) &= (j, j)
 \end{aligned} \tag{2.9}$$

Дерево $TreeImage$ (универсум (схема 2) для всех изображений, имеющих не более n дуг, и k вариантов дуг и связей дуг) строится по аналогии с деревом $Tree$ для декартова произведения

$$Arc_1 \times Rel_1 \times Arc_2 \times Rel_2 \times \dots \times Rel_{n-1} \times Arc_n$$

Соглашения по представлению элементов схемы 2 следующие:

1. для элемента $\alpha_{\mu,\chi}^{\theta}$ - число θ является позицией на этаже схемы (номер клетки в строке); $\alpha_{\mu,\chi}^{\theta}$ является χ -ым элементом из множества Arc_{μ} или Rel_{μ} ;
2. числа m_i , где i - номер этажа, равны $k^i - k$; число $t = k^{2n-1} - k$, отметим, что данные числа имеют чисто технический характер и уменьшают громоздкость выражений, стоящих в конце строк схемы 2.

Очевидно, что схема 2 содержит все изображения, имеющие не более n дуг, и k вариантов дуг и связей дуг. Структуру дерева на схеме 2 будем задавать отношениями $Par_{arc}(x, y)$, $Brot_{arc}(x, y)$ для дуг, и $Par_{rel}(x, y)$ для связей дуг.

Отношение $Par_{arc}(x, y)$ задает отношение «родитель-потомок» на декартовом произведении $Arc \times Rel$, например, $Par_{arc}(ar_{i,j}^2, re_{i,w}^{k(j-1)+w})$, где $1 \leq w \leq k$. Отношение $Par_{arc}(x, y)$ связывает элементы, расположенные на соседних этажах, и может быть определено строго математически, а именно, $Par_{arc}(\alpha_{\mu,\chi}^{\theta}, \beta_{\zeta,\eta}^{\gamma})$ тогда и только тогда, когда

1. $\mu = \zeta$ или $\mu + 1 = \zeta$
2. $(\theta - 1)k \leq \zeta \leq (\theta - 1)k + k - 1$

Отношение $Brot_{arc}(x, y)$ задает отношение «быть братом» на декартовом произведении $Arc \times Arc$. Отношение $Brot_{arc}(x, y)$ связывает элементы, расположенные на одном этаже и связанные с одним элементом верхнего этажа отношением «родитель-потомок», и может быть определено строго математически, а именно, $Brot_{arc}(\alpha_{\mu,\chi}^\theta, \beta_{\zeta,\eta}^\gamma)$ тогда и только тогда, когда

1. $\alpha = \beta$;
2. $\mu = \zeta$;
3. $\theta < \zeta$ и $\theta - \zeta < k$, а также $[\theta/k] > 0$, где операция $[]$ остаток от деления.

Отношение Par_{rel} задается по аналогии, на декартовом произведении $Rel \times Arc$.

Интерпретация ξ произвольного связного изображения 2.3 (в дальнейшем термин «изображение» будет означать только «изображение, имеющие не более n дуг, и k вариантов дуг и связей дуг», если, конечно, не оговорено противное), где

$$Arc = \{ar_1, ar_2, \dots, ar_w\}, w \leq n, Rel = \{re_1, re_2, \dots, re_t\} \quad (2.10)$$

на дереве $TreeImage$ производиться по следующей схеме

Основание индукции. Пусть $i = 1$ и $Sector(ar_1) = (j, j)$ и

$$Rrel_1 = \{re_i \mid R(re_i, Ar_1, Ar_2), Ar_1 = ar_1 \vee Ar_2 = ar_1\}$$

$$\begin{aligned} Arr_1 = \{aar_i \mid & R(re_i, Ar_1, Ar_2), re_i \in Rrel_1, \\ & (Ar_1 = ar_1 \wedge Ar_2 = aar_i) \vee (Ar_2 = ar_1 \wedge Ar_1 = aar_i)\}. \end{aligned}$$

Тогда полагаем $\xi(ar_1) = ar_{1,j}^j$, $\xi(re_i) = re_{1,v}^{(j-1)*k+v}$, где $re_i \in Rrel_1$, $Angle(re_i) = Angle(re_{1,v}) = v$. Если $aar_i \in Arr_1$, то

$$\xi(aar_i) = ar_{2,e}^{(d-1)*k+e},$$

где d – позиция элемента $\xi(re_i)$ (т.е. $d = (j-1)*k + v$, $Sector(aar_i) = (e, e)$).

Отметим, что $Par_{arc}(\xi(ar_1), \xi(re_i))$, $Par_{rel}(\xi(re_i), \xi(aar_i))$, а также для любых aar_i и aar_j из Arr_1 выполняется $Brot_{arc}(\xi(aar_i), \xi(aar_j))$.

Интерпретация ξ продолжается для множества дуг

$$Arc_1 = Arc \setminus (\{ar1\} \cup Arr_1),$$

множества связей дуг

$$Rel_1 = Rel \setminus Rrel_1.$$

Замечание 2.12. Важнейшим моментом построения отношений $Par_{arc}(x, y)$, $Brot_{arc}(x, y)$ и $Par_{rel}(x, y)$ на схеме 2 является их конструктивизм (эффективная вычислимость за один шаг) и это свойство сохраняется при построении интерпретации ξ , как для основания индукции (так и для индукционного шага, что будет показано ниже).

Индукционный шаг. Пусть после i -го шага получены непустые множества дуг $Arr_i = \{ar_{i_1}, ar_{i_2}, \dots, ar_{i_w}\}$, $Arc_i = Arc_{i-1} \setminus Arr_i$, множества связей дуг $Rrel_i$, $Rel_i = Rel_{i-1} \setminus Rrel_i$, причем по аналогии с основанием индукции, дуги из множества $\xi(Arr_i)$ располагаются на $2*i + 1$ этаже схемы 2, связи дуг из множества $\xi(Rrel_i)$ располагаются на $2*i$ этаже схемы 2.

По алгоритму основания индукции будем проводить построения для каждой дуги $ar_\alpha \in Arr_i$ такой, что ar_α не принадлежит объединению $\{ar_1\} \cup Arr_1 \cup \dots \cup Arr_{i-1}$. Пусть $Sector(ar_\alpha) = (j, j)$ и

$$Rrel_{i+1,\alpha} = \{re_s \mid R(re_s, Ar_1, Ar_2), re_s \in Rel_i, Ar_1 = ar_\alpha \vee Ar_2 = ar_\alpha\}$$

$$\begin{aligned} Arr_{i+1,\alpha} = & \{aar_u \mid R(re_s, Ar_1, Ar_2), \\ & (Ar_1 = ar_\alpha \wedge Ar_2 = aar_u) \vee (Ar_2 = ar_\alpha \wedge Ar_1 = aar_u)\} \end{aligned}$$

Пусть $\xi(ar_\alpha) = ar_{i+1,j}^\beta$. Тогда $\xi(re_s) = re_{i+1,v}^{(\beta-1)*k+v}$, где $re_s \in Rrel_{i+1,\alpha}$,

$$Angle(res) = Angle(re_{i+1,v}^{(\beta-1)*k+v}) = v.$$

Если $aar_u \in Arr_{i+1,\alpha}$, то $\xi(aar_u) = ar_{2,e}^{(d-1)*k+e}$, где d – позиция элемента $\xi(re_s)$ (т.е. $d = (\beta - 1) * k + v$), $Sector(aar_u) = (e, e)$

Отметим, что $Par_{arc}(\xi(ar_\alpha), \xi(re_s))$, $Par_{rel}(\xi(re_s), \xi(aar_u))$, а также для любых aar_i и aar_j из $Arr_{i+1,\alpha}$ выполняется $Brot_{arc}(\xi(aar_i), \xi(aar_j))$.

Полагаем множество дуг Arr_{i+1} равным объединению всех $Arr_{i+1\alpha}$, где ar_α произвольная дуга из множества Arr_i такая, что ar_α не принадлежит объединению $\{ar_1\} \cup Arr_1 \cup \dots \cup Arr_{i-1}$, также множество связей дуг $Rrel_{i+1}$ полагаем равным объединению всех $Rrel_{i+1\alpha}$, соответствующие произвольным дугам ar_α из множества Arr_i (смотри, выше).

Интерпретация ξ продолжается для множества дуг

$$Arc_{i+1} = Arc_i \setminus Arr_i,$$

и для множества связей дуг

$$Rel_{i+1} = Rel_i \setminus Rrel_{i+1}.$$

Так как по условию интерпретируемое изображение 2.10 является связным, то процесс интерпретации ξ будет закончен не более, чем за w шагов индукции, где w – количество дуг.

Отметим, что соответствие при построении интерпретации ξ для любой дуги или связи дуг изображения 2.10 производится за один шаг, так как «связывание», соответствующего элемента схемы 2, производится вычислением одной арифметической формулы. Таким образом, доказана

Лемма 2.5.1. *Верхняя граница сложности построения интерпретации ξ для связного изображения 2.3 не превышает $O(w + t)$, где w – количество дуг, t – связей дуг.*

Теорема 2.5.1. *Пусть каждая из многоосновных а.с. R_1, R_2, \dots, R_m 2.5 имеет не более n дуг и представляет связное изображение. Тогда анализ связного изображения 2.3 имеет верхнюю границу сложности не превышающую $O(((w + t) * w) + m)$, где w – количество дуг (t – количество связей дуг) изображения 2.3, причем множества дуг и связей дуг представлены выражениями 2.10.*

Доказательство. Построим интерпретации всех многоосновных а.с. R_1, R_2, \dots, R_m на универсуме схема 2 для всех изображений, имеющих не более n дуг, и k вариантов дуг и связей дуг (сложность этой процедуры, конечно, не входит в оценку доказываемой теоремы).

Далее, пометим все вершины схемы 2 номерами многоосновных а.с., чьи элементы соответствуют этим вершинам. Каждой многоосновной а.с. R_i сопоставим пару чисел (a_i, b_i) , где a_i – количество помеченных вершин 2, соответствующих дугам, b_i – количество помеченных вершин схемы 2, соответствующих связям дуг (конечно, помеченных номером i).

Построим совокупность интерпретаций $\xi_1, \xi_2, \dots, \xi_w$ на схеме 2 (с помеченными вершинами), которые отличаются выбором первой дуги для основания индукции. А именно, интерпретация ξ_1 начинается традиционно с дуги ar_1 , интерпретация ξ_2 начинается с дуги ar_2 , и так далее. Последняя интерпретация ξ_w начинается, соответственно, с дуги ar_w .

Введем для каждой интерпретации ξ_i множество пар

$$(a_{i_1}, b_{i_1}), (a_{i_2}, b_{i_2}), \dots, (a_{i_m}, b_{i_m}) \quad (2.11)$$

где $a_{i_j}(b_{i_j})$ - количество помеченных j вершин схемы 2, соответствующих дугам (соответственно, связям дуг), полученных для интерпретации ξ_i . Если пара (a_{i_j}, b_{i_j}) равна паре (a_j, b_j) , то, таким образом, найдено изоморфное вложение j -го изображения (образца) в анализируемое изображение .

В силу леммы 2.5.1, построение каждого отображения ξ_i требует не более $w+t$ шагов и, таким образом, верхняя граница сложности поиска всех изоморфных вложений не более $O(((w+t)*w)+m)$, где «добавка» $O(m)$ возникает из-за необходимости сравнивать пары 2.11 с парами (a_j, b_j) . \square

Замечание 2.13. *Как и в случае для базовой постановки задачи, на практике универсум (схема 2) не строится а строится только его часть, состоящая из дуг и связей дуг многоосновных а.с. R_1, R_2, \dots, R_m 2.5.*

2.6 Оценка сложности анализа изображений с метрикой

Пусть многоосновные а.с.

$$\begin{aligned}\mathfrak{R}_1 &= \langle A_1, R_1, V, M; Sector, Angle, Metric, Relation \rangle \\ \mathfrak{R}_2 &= \langle A_2, R_2, V, M; Sector, Angle, Metric, Relation \rangle \\ &\dots \\ \mathfrak{R}_m &= \langle A_m, R_m, V, M; Sector, Angle, Metric, Relation \rangle\end{aligned}\tag{2.12}$$

задают искомые образцы в анализируемом изображении 2.1.

Анализ изображения 2.1 состоит в поиске всех изоморфных вложений $\mu_{i,j}$ многоосновных а.с. $\mathfrak{R}_1, \mathfrak{R}_2, \dots, \mathfrak{R}_m$ в многоосновную а.с. \mathfrak{M} 2.1, т.е. изоморфное вложение $\mu_{i,j} : \mathfrak{R}_i \rightarrow \mathfrak{M}$ состоит из инъективных отображений

$$\mu_{i,j} : A_i \rightarrow A$$

$$\mu_{i,j} : R_i \rightarrow R$$

таких, что:

- a) если $\mu_{i,j}(a') = a$, где $a' \in A_i, a \in A$ и

$$Sector(a') = [c'_{min}, c'_{max}],$$

$$Sector(a) = [c_{min}, c_{max}],$$

то $[c_{min}, c_{max}] \in [c'_{min}, c'_{max}]$;

- б) если $\mu_{i,j}(a') = a$, где $a' \in A_i, a \in A$ и

$$Metric(a') = [c'_{min}, c'_{max}],$$

$$Metric(a) = [c_{min}, c_{max}],$$

то $[c_{min}, c_{max}] \in [c'_{min}, c'_{max}]$;

в) если $\mu_{i,j}(r') = r$, где $r' \in R_i$, $r \in R$ и

$$Angle(r') = [c'_{min}, c'_{max}],$$

$$Angle(r) = [c_{min}, c_{max}],$$

то $[c_{min}, c_{max}] \in [c'_{min}, c'_{max}]$;

г) если $Relation(r, a_1, a_2)$, где $r \in R_i$, $a_1, a_2 \in A_i$, то

$$Relation(\mu_{i,j}(r), \mu_{i,j}(a_1), \mu_{i,j}(a_2))$$

Определим универсум для изображений, имеющих не более n дуг, и не более k связей дуг (это ограничение ни сколько не повлияет на результат, но немного упростит нам форму записи).

Пусть A_1, A_2, \dots, A_m – множества дуг всех характеристик (образцов), т.е.

$$\begin{aligned} A_1 &= \{a_{1,1}, a_{1,2}, \dots, a_{1,n}\}; \\ A_2 &= \{a_{2,1}, a_{2,2}, \dots, a_{2,n}\}; \\ &\dots \\ A_m &= \{a_{m,1}, a_{m,2}, \dots, a_{m,n}\}; \end{aligned} \tag{2.13}$$

Далее пусть R_1, R_2, \dots, R_{m-1} – множества связей дуг всех характеристик, т.е.

$$\begin{aligned} R_1 &= \{r_{1,1}, r_{1,2}, \dots, r_{1,k}\}; \\ R_2 &= \{r_{2,1}, r_{2,2}, \dots, r_{2,k}\}; \\ &\dots \\ R_{n-1} &= \{r_{n-1,1}, r_{n-1,2}, \dots, r_{n-1,k}\}; \end{aligned} \tag{2.14}$$

Таким образом наше дерево соответствует следующему декартову произведению:

$$G_1 \times M_1 \times V_1 \times G_2 \times M_2 \times \dots \times V_{n-1} \times G_1 \times M_1 \tag{2.15}$$

где

$$\begin{aligned} G_i &= \{Angle(a_{i,1}), Angle(a_{i,2}), \dots, Angle(a_{i,n})\} \\ M_i &= \{Metric(a_{i,1}), Metric(a_{i,2}), \dots, Metric(a_{i,n})\} \\ V_i &= \{Sector(r_{i,1}), Sector(r_{i,2}), \dots, Sector(r_{i,k})\} \end{aligned} \tag{2.16}$$

Очевидно, что 2.15 содержит все изображения, имеющие не более n дуг, и k связей дуг. Каждый уровень дерева представляет собой набор узлов, значения которых принадлежат одному из множеств: V_{arc} , M либо V_{rel} , где $V_{arc} \subseteq V$ – множество допустимых значений секторов дуг, $V_{rel} \subseteq V$ – множество допустимых углов соединения связей дуг. У всякого дерева изображений значения узлов первого уровня принадлежат V_{arc} , значения узлов последнего уровня принадлежат M .

Однако работать с таким деревом достаточно сложно, намного удобнее редуцировать третий уровень и использовать «двух-слойную» (состоящую из двух основных множеств) модель дерева. Так как множества значения углов и относительных величин ограниченно, мы можем упорядочить множества дуг следующим образом.

Пусть $A = \{a_1, \dots, a_k, \dots, a_n\}$ – множество всевозможных дуг

$$Sector(a_k) = g_i, 0 < i < |V|, V = \{g_0, \dots, g_{|V|-1}\}$$

$$Metric(a_k) = m_j, 0 < j < |M|, M = \{m_0, \dots, m_{|M|-1}\}$$

тогда $k = i * |M| + j$.

Таким образом мы сможем оперировать двумя основными множествами: множеством дуг A и множеством связей R .

$$|A| = |V_{rel}| * |M|$$

$$|R| = |V_{arc}|$$

$$V_{rel}, V_{arc} \subseteq V$$

2.7 Масштабные ряды

Расширим базовую формализацию, добавив в качестве основного вполне упорядоченное множество *ScaleLine*, которое соответствует масштабному ряду (аналог масштабного ряда топографических карт 1:500, 1:1000 и т.д.), а также два дополнительных отношения: *Scale*, которое связывает дуги с элементами

i -являющейся образом arc при сжатии изображения, т.е. при переходе к следующему (большему на единицу) элементу масштабного ряда.

При практической реализации удобно считать вполне упорядоченное множество $ScaleLine$ отрезком натуральных чисел $[1, \dots, n]$ (это представление будет использоваться в статье для упрощения обозначений). В частности, поэтому будем использовать ниже конструкции вида: $s1 = s2 - 1$ и $s2 = s1 + 1$, где $s1, s2$ принадлежат множеству $ScaleLine$.

Предполагаем, что анализируемое изображение и образцы имеют представления для всех элементов масштабного ряда $ScaleLine$. Таким образом, совокупность всех дуг A четырехосновной а.с. 2.1 может быть представлено в виде

$$A = A_1 \bigcup A_2 \bigcup \dots \bigcup A_n \quad (2.17)$$

где множества A_i состоят из дуг, относящихся к i -му элементу масштабного ряда.

Определение 2.7.1. В дальнейшем множества дуг A_i будем называть i -ой компонентой изображения.

Замечание 2.14. Для общего подхода, охватывающего все варианты сжатия изображения, включая рассмотрение под углом (что, конечно, не соответствует генерализации для топографических карт, где используется ортогональная проекция, но весьма важно для практики, достаточно вспомнить о зрении живых организмов, где обрабатываются зрительные образы правого и левого глаза), представление масштабного ряда $ScaleLine$ отрезком натуральных чисел $[1, \dots, n]$ или даже частично упорядоченными множествами недостаточно. В дальнейшем для формализации масштабных рядов будут использоваться многоосновные а.с., включающие частично упорядоченные множества и дополнительные множества с определенными на них отношениями.

Первое отношение

$$Scale : \varsigma \in A \times ScaleLine$$

определяет принадлежность дуги отрезку масштабного ряда $ScaleLine$, т.е.

$$(Scale(a, s_1) \wedge Scale(a, s_2) \wedge s_1 \leq s \leq s_2) \rightarrow Scale(a, s)$$

Как отмечалось выше, отношение *Scale* обладает следующим основным свойством. Пусть

$$A_i = \{arc_1, arc_2, \dots, arc_r\}$$

совокупность всех дуг i -го множества из объединения 2.17. Тогда всегда $Scale(arc_k, i)$ для всех k .

Другим основным свойством отношения *Relation* является следующее: *Relation* может связывать только дуги, принадлежащие одному элементу масштабного ряда, т.е., если $Relation(rel, a_1, a_2)$, то существует элемент s из *ScaleLine* такой, что $Scale(a_1, s), Scale(a_2, s)$.

Второе отношение

$$Include : \zeta \in A \times A$$

определяет вложение дуги a_1 в дугу a_2 большего элемента масштабного ряда, т.е., если $Include(a_1, a_2)$, то существует s из *ScaleLine* такой, что $Scale(a_2, s), Scale(a_1, s)$ и $Scale(a_1, s - 1)$.

Определим основные свойства отношения вложения *Include*, которые математически формулируются следующим образом:

1. $(Include(a_1, a_2) \wedge Include(a_1, a_3)) \rightarrow a_2 = a_3;$
2. $(s_1 = s_2 - 1 \wedge Scale(a_1, s_1)) \rightarrow \exists a_2 : Scale(a_2, s_2) \wedge Include(a_1, a_2);$
3. $(s_1 = s_2 - 1 \wedge Scale(a_1, s_2)) \rightarrow \exists a_2 : Scale(a_2, s_1) \wedge Include(a_2, a_1);$

Переформулируем основные свойства отношения вложения *Include* более понятным образом, через отображения F_i определяемые отношением вложения *Include* из множеств A_i в множества A_{i+1} . Определим последовательность функций сжатия $F_i : A_i \rightarrow A_{i+1}$ следующим образом:

$$F_i(arc1) = arc2 \leftrightarrow Include(arc1, arc2) \quad (2.18)$$

где дуга arc_1 из множества A_i , а дуга arc_2 из множества A_{i+1} .

Лемма 2.7.1. *Функция сжатия F_i , заданная формулой 2.18, всюду определена, однозначна (т.е. отображение) и является эпиморфизмом. Доказательство следует из определения отношения *Include*, и свойств 1,2,3 отношения *Include*.*

Замечание 2.15. В силу леммы 2.7.1 в дальнейшем во многих случаях вместо отношения *Include* будем использовать совокупность отображений F_i , которые будем называть сжатиями при переходе от i -го к $i+1$ -ому элементу масштабного ряда, где $i \geq 1$ и $i < n$.

Следующее основное свойство определяет сохранение отношения связности *Relation* дуг при сжатии,

$$(F_i(arc_1) = arc_3 \wedge F_i(arc_2) = arc_4 \wedge arc_3 \neq arc_4 \wedge Relation(rel, arc_1, arc_2)) \rightarrow Relation(rel,$$

Определение 2.7.2. Подмножество дуг $B = \{b_1, b_2, \dots, b_s\}$ и компоненты A_i 2.17 назовем связным, если для любых двух дуг b_i, b_j , существует последовательность дуг c_1, c_2, \dots, c_u из B такая, что каждая пара дуг c_v, c_{v+1} являются связанными, $c_1 = b_i$, $c_u = b_j$.

Прежде всего, необходимо доказать, что прообраз и образ связного множества остается связным, что и показывает следующая

Лемма 2.7.2. Пусть подмножество дуг $B = \{b_1, b_2, \dots, b_s\}$ i -компоненты A_i 2.17 является связным, причем $i > 1$. Далее, пусть B_1 (B_2) подмножества дуг $i-1$ компоненты A_{i-1} (соответственно, $i+1$ компоненты A_{i+1}), которые будем называть прообразом (соответственно, образом) подмножества дуг B относительно функций сжатия 2.18, которые строго определим следующим образом:

$$B_1 = \{b \mid b \in A_{i-1}, F_{i-1}(b) \in B\}$$

$$(соответственно, B_2 = \{b \mid b \in A_{i+1}, b = F_i(b_j) \in B\})$$

Тогда подмножество дуг B_1 (соответственно, B_2) является связным.

Доказательство. Пусть d_1, d_2 произвольные дуги из B_1 (соответственно, B_2). Тогда по определению B_1 (соответственно, B_2) в подмножестве дуг B существуют дуги b_l, b_k такие, что $F_{i-1}(d_1) = b_l, F_{i-1}(d_2) = b_k$ (соответственно, $F_i(b_l) = d_1, F_i(b_k) = d_2$).

Так как подмножество дуг B является связным, то существует последовательность дуг c_1, c_2, \dots, c_u из B такая, что каждая пара дуг c_v, c_{v+1} являются связанными, $c_1 = b_l, c_u = b_k$.

Определим последовательность дуг e_1, e_2, \dots, e_u из B_1 (соответственно, B_2) следующим образом:

1. $e_1 = d_l$, $e_u = d_2$, отметим, что $F_{i-1}(e_1) = c_1$, $F_{i-1}(e_u) = c_u$ (соответственно, $F_i(c_1) = e_1$, $F_i(b_k) = d_2$)
2. $F_{i-1}(e_j) = c_j$ (соответственно, $F_i(c_j) = e_j$) для всех j из отрезка $[2, u-1]$.

Тогда последовательность дуг e_1, e_2, \dots, e_u из B_1 (соответственно, B_2) такая, что каждая пара дуг e_v, t_{v+1} являются связанными по основному свойству отношения $Relation(rel, arc_1, arc_2)$, и $e_1 = d_l$, $e_u = d_2$, что показывает связность подмножеств дуг B_1 (соответственно, B_2). \square

Важным является следующее

Следствие. Прообраз подмножества, состоящего из одной дуги, относительно функций сжатия F_i 2.18, является связным.

2.7.1 Результаты

Теперь можем приступить к доказательству основной цели работы: того, что, если сжатый образец изоморфно вложен в сжатое изображение (предполагается, что образцы и изображения используют один и тот же масштабный ряд), то исходный образец может быть изоморфно вложен в исходное изображение, тогда и только тогда, когда прообраз каждой дуги образца может быть изоморфно вложен в прообраз соответствующей дуги изображения.

Во избежание громоздкого «всеобщего случая» будем рассматривать изоморфное вложение одного образца, представленного многоосновной а.с.

$$S = \langle A_s, R_s, V_s, M_s; Sector, Angle, Relation \rangle, \quad (2.19)$$

в изображение, представленное многоосновной а.с.

$$W = \langle A_w, R_w, V_w, M_w; Sector, Angle, Relation \rangle \quad (2.20)$$

$$A_s = A_{s1} \cup A_{s2} \cup \dots \cup A_{sn} (A_w = A_{w1} \cup A_{w2} \cup \dots \cup A_{wn}) \quad (2.21)$$

где множества A_{s_j} (соответственно, A_{w_j}) состоят из дуг, относящихся к j -му элементу масштабного ряда 2.1. Будем считать, что последовательность функций вложения

$$F_i : A_i \rightarrow A_{i+1} \quad (2.22)$$

определенна на множествах дуг A_s и A_w . Тогда совокупность дуг A_{s_2} (соответственно, A_{w_2}) является сжатием совокупности дуг A_{s_1} (соответственно, A_{w_1}) и $F_1(A_{s_1}) = A_{s_2}$ (соответственно, $F_1(A_{w_1}) = A_{w_2}$).

Определение 2.7.3. Пусть $\zeta : A_{s_2} \rightarrow A_{w_2}$ произвольное отображение. Отображение $\xi : A_{s_1} \rightarrow A_{w_1}$ назовем экстраполяцией отображения ζ , если для любой дуги arc из A_{s_1} выполняется: пусть $\xi(arc) = arc_1$, тогда $\zeta(F_1(arc)) = F_1(arc_1)$.

Теорема 2.7.1. Пусть $\zeta : A_{s_2} \rightarrow A_{w_2}$ изоморфное вложение сжатой совокупности дуг A_{s_2} из S 2.18 в сжатую совокупность дуг A_{w_2} из W 2.21. Тогда изоморфное вложение ζ может быть экстраполировано до изоморфного вложения $\xi : A_{s_1} \rightarrow A_{w_1}$ тогда и только тогда, если для любой дуги arc из A_{s_2} прообраз (относительно функции сжатия F_1) дуги arc может быть изоморфно вложен в прообраз (относительно функции сжатия F_1) дуги $\zeta(arc)$.

Доказательство. Предположим, что прообраз каждой дуги $F_{i-1}(arc)$ образца A_{s_2} может быть изоморфно вложен в прообраз соответствующей дуги изображения $F_{i-1}(\zeta(arc))$ изображения A_{w_2} . Обозначим через ξ_{arc} такое изоморфное вложение, т.е. $\xi_{arc} : F_{i-1}(arc) \rightarrow F_{i-1}(\zeta(arc))$. Положим вложение $\xi = \xi_{arc_1} \cup \xi_{arc_2} \cup \dots \cup \xi_{arc_L}$, где $A_{s_2} = \{arc_1, arc_2, \dots, arc_L\}$. Тогда ξ является изоморфным вложением, причем экстраполированным с изоморфным вложением

$\zeta : A_{s_2} \rightarrow A_{w_2}$ (следует из того, что $\xi_{arc} : F_{i-1}(arc) \rightarrow F_{i-1}(\zeta(arc))$) и достаточность для теоремы доказана. Докажем необходимость. Пусть $\zeta : A_{s_1} \rightarrow A_{w_1}$ является изоморфным вложением из совокупности дуг образца A_{s_1} в совокупность дуг изображения A_{w_1} экстраполированным с изоморфным вложением $\zeta : A_{s_2} \rightarrow A_{w_2}$. Тогда ограничение вложения ξ на $F_{i-1}(arc)$ будет изоморфным вложением в $F_{i-1}(\zeta(arc))$ для всех дуг arc из совокупности A_{s_2} . \square

Замечание 2.16. Конечно, программная реализация масштабных рядов и обеспечение эффективной работы функций сжатия F_i 2.22 (а также обратных

функций F_{i-1}) требует значительных усилий, которые должны быть «вознаграждены» повышением эффективности поиска изоморфных вложений образцов, что и будет показано во второй части данной работы.

Пусть каждая из многоосновных а.с. R_1, R_2, \dots, R_m вида (1) имеет не более n дуг и представляет связное изображение. Тогда анализ связного изображения, представленного многоосновной а.с. D , имеет верхнюю границу сложности не превышающую $O(((w + t)^* w) + m)$, где $w(t)$ – количество дуг (соответственно, связей дуг) изображения, представленного многоосновной а.с. D . Пусть многоосновные а.с. $Sh = < Ash, Rs, Vs, Ms; Sector, Angle, Relation >$, (13)

$(Wh = < Awh, Rw, Vw, Mw; Sector, Angle, Relation >)$ (14) получены из многоосновной а.с. S (9) (соответственно, W (10)) уменьшением множества дуг до относящихся только к h - му элементу масштабного ряда (11), где h изменяется от 1 до n . Положим также, что количество дуг в множестве Awh равно dh . Для наших целей будет удобна теорема А в следующей формулировке. Теорема Б. Построение изоморфного вложения многоосновной а.с. Sh (13) в многоосновную а.с. Wh (14) имеет верхнюю границу сложности не превышающую $O((dh + t)^* dh)$, где $dh(t)$ – количество дуг (соответственно, связей дуг) многоосновной а.с. Wh (14). Замечание 4. Возможность использования одной и той же совокупности связей Rs (Rw) для всей совокупности многоосновных а.с. Sh (соответственно, Wh) следует по основному свойству (*), которое сохраняет отношение связности $Relation$ дуг при сжатии. Хотя, конечно, при сжатии некоторые связи дуг могут становиться ненужными и учет этого, улучшил бы оценку теоремы Б. Замечание 5. 1) Оценка сложности вычислена с большим запасом, т.к., например, в качестве коэффициента d_2 можно было взять количество дуг именно As_2 , а не количество дуг Aw_2 . 2) Совмещение деревьев интерпретаций многоосновных а.с. S_1 и S_2 существенно улучшило бы оценки последней теоремы, но имеет большие технические трудности и не может быть представлена в этой работе. Теорема 3. Использование изоморфного вложения сжатого образца, представленного многоосновной а.с. S_2 , уменьшает сложность построения изоморфного вложения исходного образца, представленного многоосновной а.с. S_1 . Доказательство. Прямое применение оценок Теоремы Б дает верхнюю границу сложности изоморфного вложения исходного образца, представленного многоосновной а.с. S_1 не превышающую $O((d_1 + t)^* d_1)$, где $d_1 = d_2 * E_1$, а верхняя

граница сложности изоморфного вложения исходного образца, представленного многоосновной а.с. S1 с использование изоморфного вложения сжатого образца, представленного многоосновной а.с. S2 дает оценку $O((d2 + t)^* d2) + O((E1 + t)^* d2) = O((d2 + E1 + 2 * t) * d2)$ заведомо меньшую, так как коэффициент сжатия на практике для более или менее сложных изображений всегда более 10. Теорема доказана.

4. Заключение. Программная реализация предложенного подхода будет иметь существенно большую эффективность, чем приведенные оценки Теоремы 3. Это связано с тем, что использование изоморфного вложения сжатых образцов предполагает: 1) Совмещение деревьев интерпретаций для всего масштабного ряда образцов, что существенно облегчает проверку изоморфной вложимости прообразов дуг относительно функций сжатия F_i . 2) Исключение «лишних» дуг при сжатии, т.е. параметр t оценок эффективности для сжатых образцов (а тем более для прообразов дуг) будет заведомо меньшим.

Глава 3

Приложения

3.1 Распознавание символов

В рамках разработки алгоритмов распознавания была спроектирована база данных для хранения информации об образцах и разработан комплекс программ для проверки практической применимости разработанных методов.

База реализована с использованием СУБД MSSQL. Проект БД приведен в приложении А.

Комплекс ПО состоит из трех программ:

1. Конвертор растровых изображений
2. Браузер для базы данных
3. Интерпретатор

Все три программы написаны на языке C++. В качестве GUI используется Borland VLC. Программы используют общую базу данных, доступ к которой осуществляется через ODBC, что позволяет использовать любую совместимую СУБД.

3.1.1 Конвертор растровых изображений

Осуществляет преобразование растрового изображения в а.с. вида 2.1. В программе присутствует простой графический редактор.

Нарисованное изображение можно преобразовать в систему дуг и связей дуг.

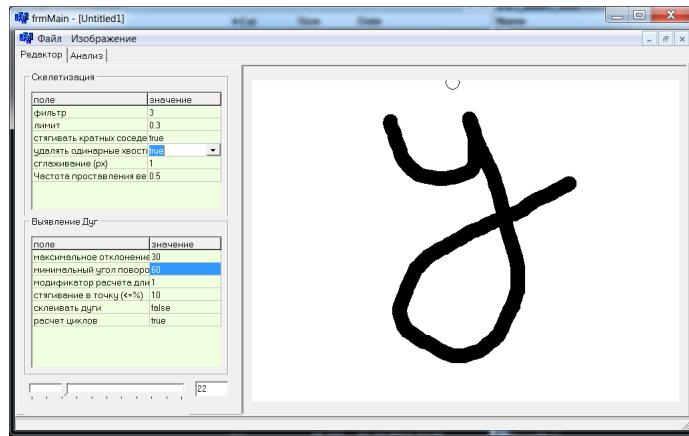
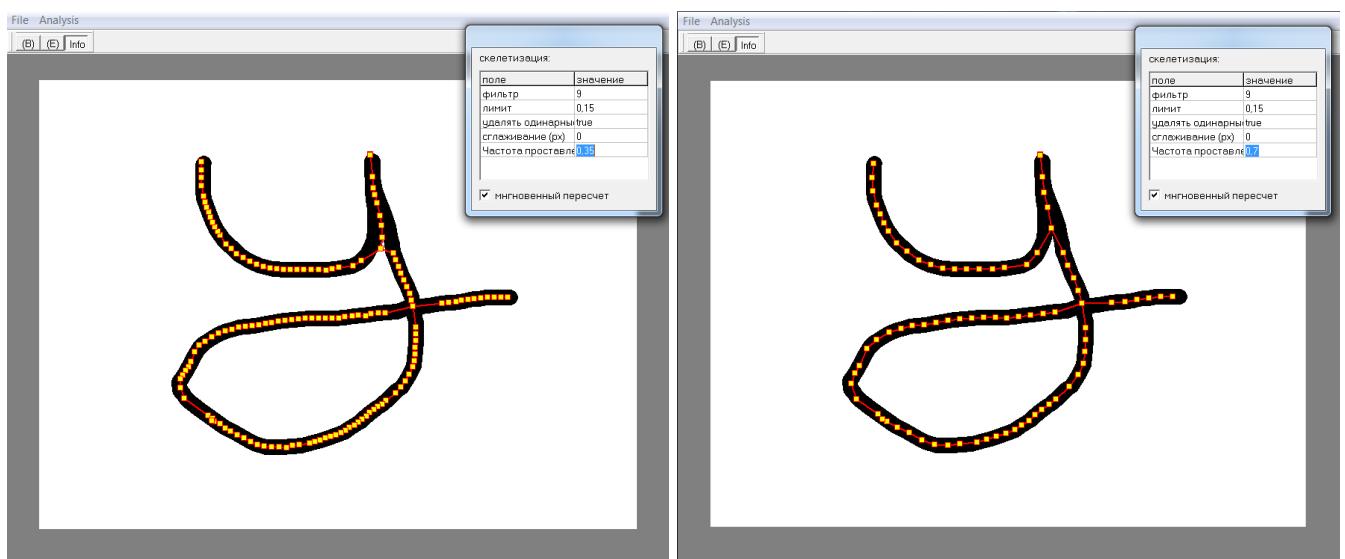


Рисунок 3.1: Графический редактор



(a) шаг: 0.35 от ширины волны

(b) шаг: 0.75 от ширины волны

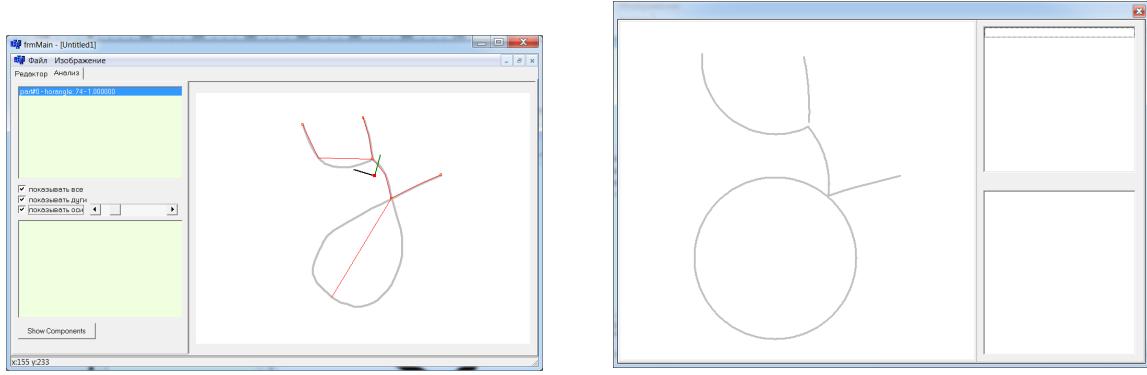
Рисунок 3.2: Разный шаг аппроксимации графа

В систему встроена система простого обучения. Обучение осуществляется путем сопоставления соответствующих дуг. Как результат обучения в БД в качестве изображения отправляется система, где всякой дуге сопоставляется не конкретное значение сектора, а некий промежуток значений.

Преобразованное изображение можно сохранить в БД в качестве нового образца.

3.1.2 Браузер для БД

Используется для просмотра образцов хранящихся в БД. Наглядно отображает связи в изображении, и умеет пересчитывать их относительно выбранной дуги. Интерфейс программы можно посмотреть на рис. 3.5.



(а) Преобразованное изображение, нарисованное поверх исходного

(б) Преобразованное изображение, нарисованное по набору дуг и связей между ними

Рисунок 3.3: Преобразованное изображение

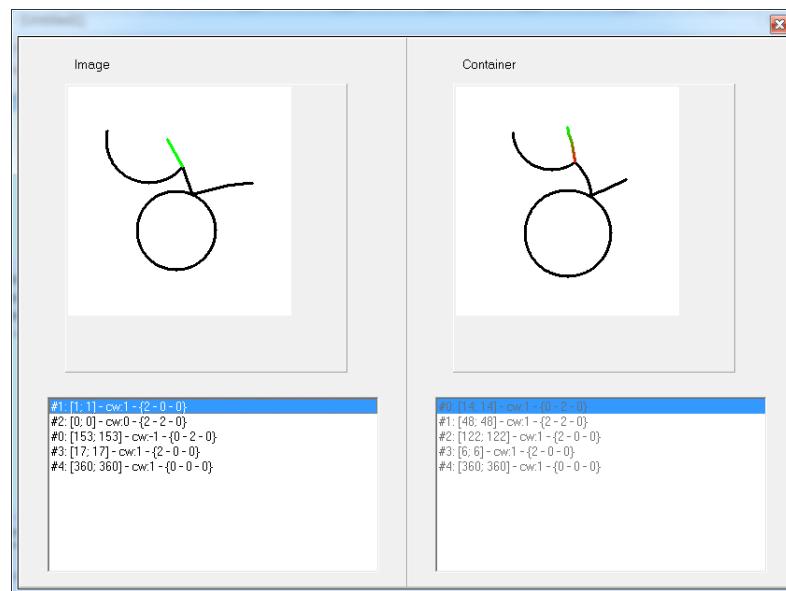
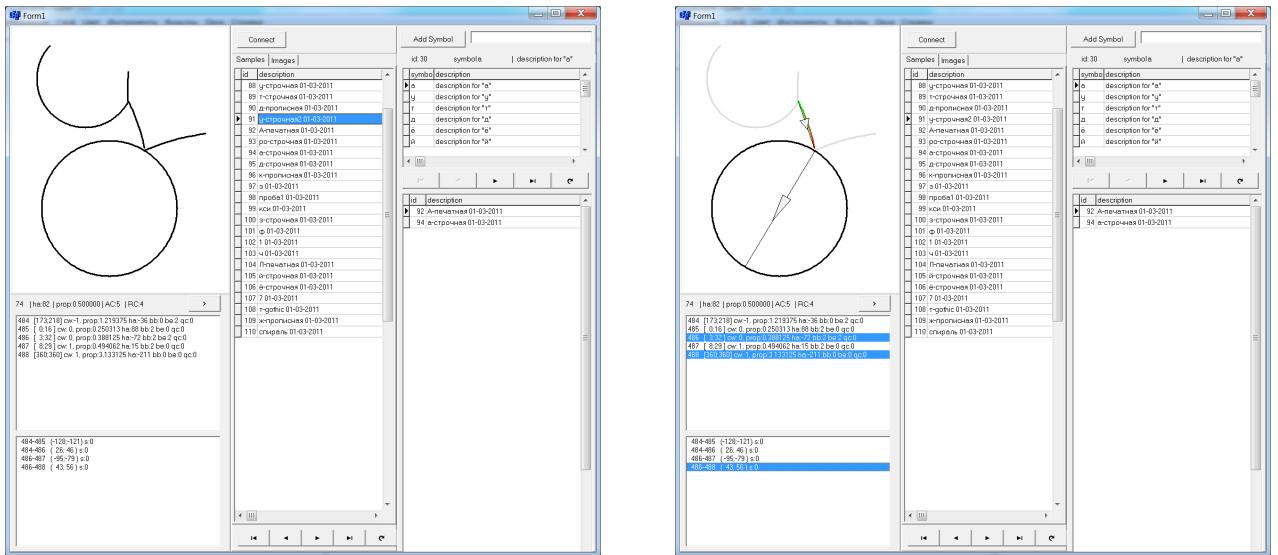


Рисунок 3.4: Обучение системы

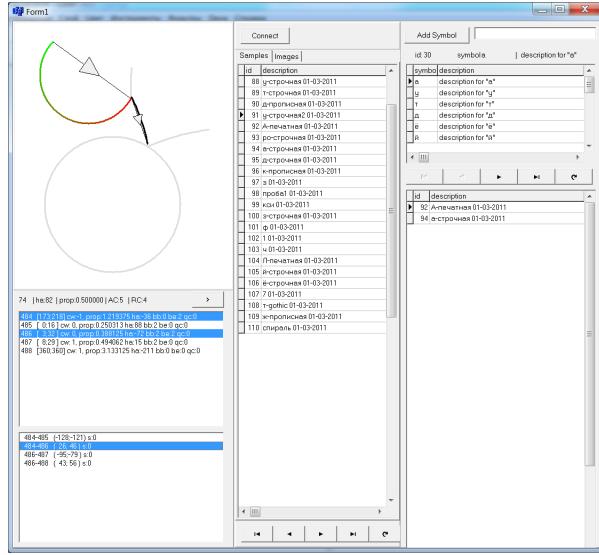
3.1.3 Интерпретатор

Предназначен для сопоставления анализируемого изображения с анализируемым образцом. В интерпретаторе реализованы алгоритмы, рассмотренные в параграфе 2.5. Программа считывает данные из БД образцов в локальную БД и осуществляет сопоставление обработанного изображениями с образцами хранящимися в БД. Интерфейс изображение представлен на рис. 3.6.



(а) Изображение построенное по набору дуг и связей между ними

(б) Связь между дугой и круговой дугой (360 градусов)



(с) Связь между двумя дугами

Рисунок 3.5: Преобразованное изображение

3.2 Оценка устойчивости битумных эмульсий

3.2.1 Введение

В настоящее время трудно назвать область строительства, где бы не применялись эмульсии. Они используются в дорожном и в гражданском строительстве в качестве связующих с различными наполнителями, а также в качестве гидроизоляционных и лакокрасочных материалов. При любых технологиях использования эмульсий мы сталкиваемся с одними и теми же проблемами, касающимися подбора состава, приготовления, определения физико-механических ха-

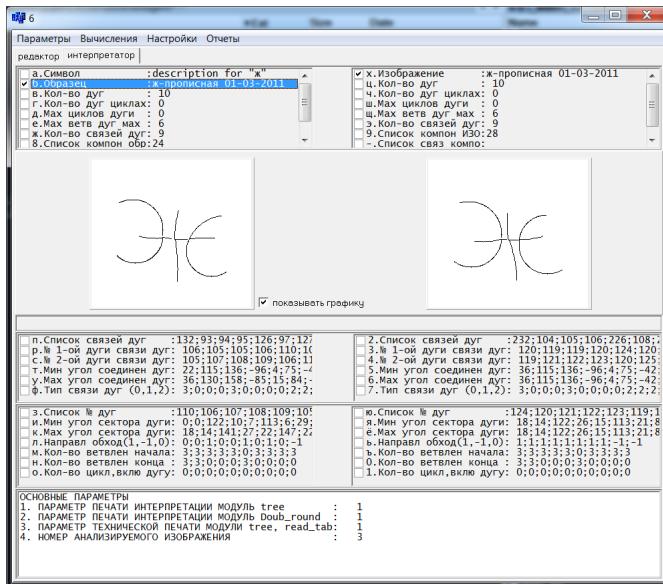


Рисунок 3.6: Интерфейс интерпретатора

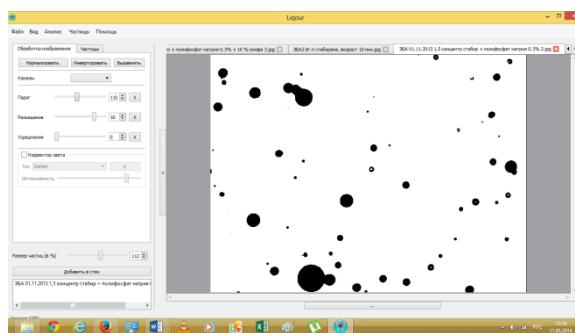
рактеристик, стабильности, контроля распада эмульсий и получения продукции с необходимыми свойствами [1]. Далее мы будем рассматривать только прямые битумные и битумно-латексные эмульсии, которые являются наиболее крупнотоннажным продуктом: мировое использование составляет миллионы тонн в год.

Традиционные методы оценки свойств битумных эмульсий включают: определение содержания вяжущего с эмульгатором, определение устойчивости эмульсии при перемешивании, определение остатка на сите, определение условной вязкости, определение устойчивости при хранении, определение адгезии эмульсий с поверхностью наполнителей, определение устойчивости при транспортировке и т.п. [2]. Наряду с традиционными методами изучения качества эмульсии, во многих приложениях желательно знать более тонкие характеристики, например: функцию распределения по размерам. Эта характеристика является одним из важнейших параметров и позволяет предсказывать большинство свойств эмульсии. Обычно размер частиц оценивают с помощью определения остатка на сите с заданным размером ячейки, но такой метод позволяет оценивать только верхний предел размеров частиц эмульсии. Полная картина распределения частиц по размеру может быть измерена с использованием таких технических приёмов как рассеяние света, микроскопия с анализом изображений, или же с помощью техники электроозонирования («техники Культера» - Coulter). Точный анализ размеров частиц битумной эмульсии может решить многие про-

блемы, которые в настоящее время являются актуальными в сфере производства битумных эмульсий:

1. Влияние эмульгатора и его концентрации на размер битумных частиц эмульсии.
2. Влияние модифицирующих битум добавок на качество получаемой эмульсии.
3. Корректировка технологической схемы производства эмульсии.
4. Влияние размера битумных частиц на основные физические свойства эмульсии.

Оптическая микроскопия, как способ распределения частиц по размерам, является наиболее удобным и точным. Например, если в способе «рассеяние света» могут возникнуть проблемы с отражением света от черных поверхностей, такими являются частицы битума, то в способе микроскопии, при высоком контрасте черного цвета, напротив, можно отличить частицы от среды, в которой они находятся.



Очевидно изображения такого вида являются одним из примеров растровых контурных изображений, с несколькими контурами. В отличие от задачи распознавания символов, здесь нет необходимости анализировать скелет изображения. Куда более важную роль играет внешний контур. Разбивая контур на дуги мы можем классифицировать частицы по уровню распада:

1. одиночные
2. слипшиеся

3. распавшиеся

Большое количество распавшихся частиц является свидетельством того что смесь является неустойчивой, а следовательно некачественной.

3.2.2 Анализ эмульсий

В рамках сотрудничества с кафедрой «Автомобильных дорог» НИ ИрГТУ была разработана программа для первичной оценки качества битумных эмульсий

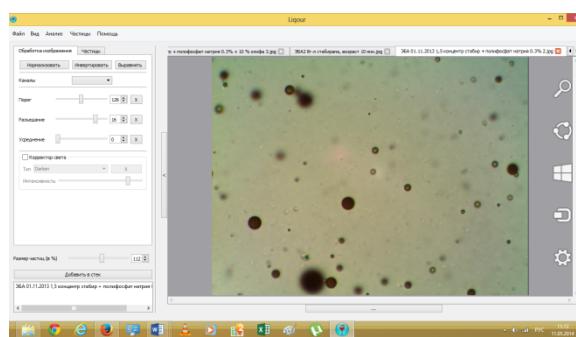


Рисунок 3.7: Скриншоты программы «анализ изображений»

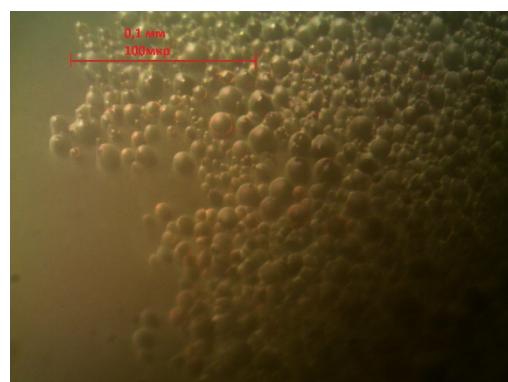


Рисунок 3.8: Концентрированная битумная анионная эмульсия в отраженном свете

На снимке программы (рис. 3.7), видно, что в проходящем свете частицы легко отличить друг от друга, но для получения такого изображения необходимо создать некоторое пространство между ними, в противном случае изображение получается как на рис. 3.8. Поэтому перед микроскопическими исследованиями образцы эмульсии распределяют небольшим количеством (концентрация от 1:100 – 1:50) в специальном стабилизирующем растворе. Необходимость данной

процедуры объясняется тем что, частицы битума могут слипаться под стеклышком и отсутствие сцепления между ними осуществляется за счет pH среды, в которой они находятся. Для анионных эмульсий это pH-щелочной, для катионных pH-кислотный.

Гибкость настроек программы позволяет редактировать изображение вручную, убирая «мнимые частицы», затемненные области и другие недочеты фотосъемки. После чего происходит автоматический подсчет частиц и построение графика функции распределения по размерам, рис. 5.

Для данной программы не требуется специализированного оборудования. Достаточно оптического микроскопа с возможностью подключения цифровой камеры и компьютер с ОС (Windows, Linux, OS).

3.2.3 Оценка качества анализа

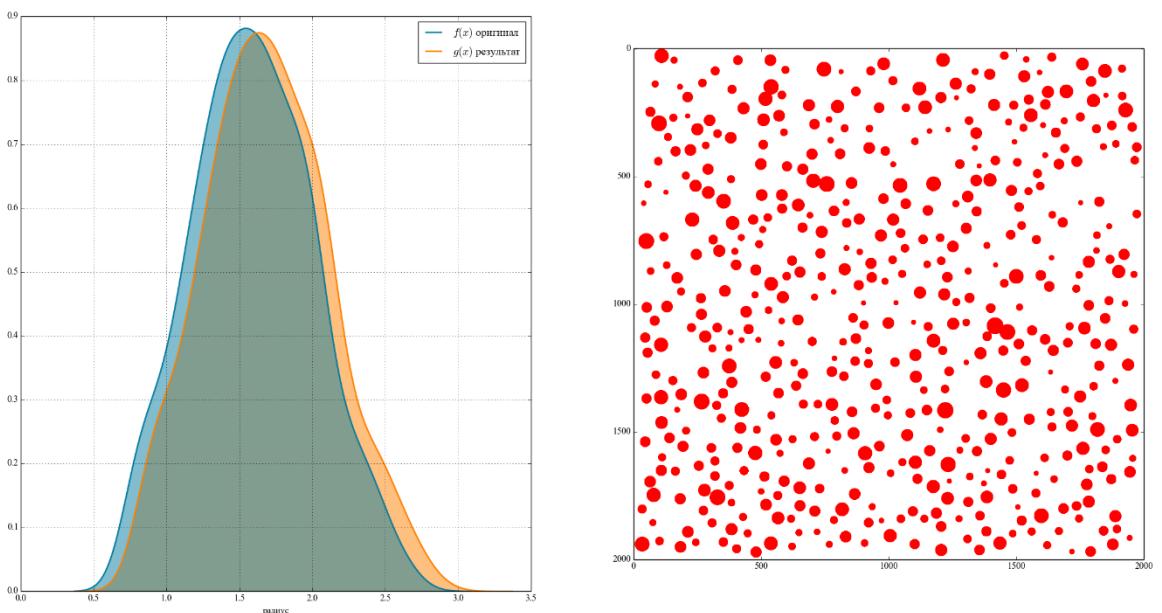


Рисунок 3.9: Результат работы программы на искусственно сгенерированных данных

На рисунке (рис. 3.9) приведены два графика: тот что сильнее смещен влево – соответствует исходным авто-сгенерированным данным, смещенный вправо соответствует распознанным данным. Формы графиков практически идентичны. Смещение, как правило, вызвано ошибками округления при распознавании объектов.

Таблица 3.1: Анализ распределения частиц на рис. 3.9

График	мин. размер частиц, мкм.	макс. размер частиц, мкм.	Среднее значение, мкм.	Ср.кв. от- клонение, мкм.	Всего ча- стиц
Исходный	0.87	2.87	1.69	0.42	500
Распознанный	0.87	2.69	1.59	0.41	493

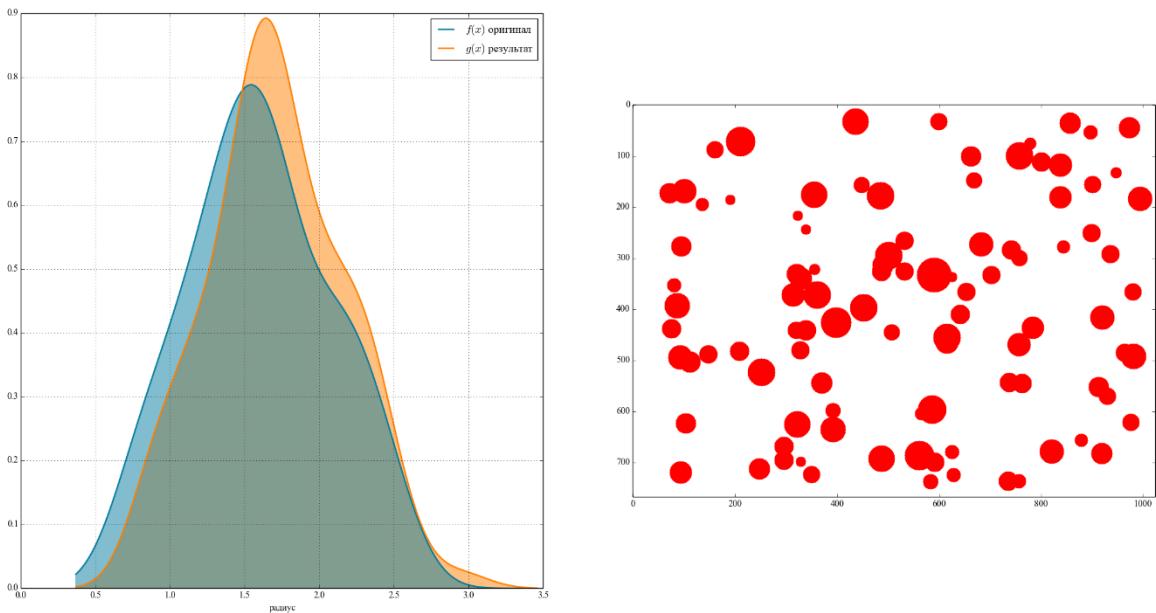


Рисунок 3.10: Проверка работы программы при наличии слипшихся частиц

Вторым фактором (первый – ошибки округления) отрицательно влияющим на качество распознавания является наличие слипшихся частиц. На рис.3.10 приведены два графика исходного (тот что более сплющен) и распознанного распределения частиц (тот что повыше). Сильное различие между графиками обусловлено тем что, при распознавании слипшиеся объекты не учитываются.

Из таблицы 3.2 видно, что, не смотря на то, что почти половина частиц не была распознана, характеристики распределения были определены достаточно точно к исходным, и погрешность составила около 4-5%.

Получение высококачественной, долговечной битумной эмульсии зависит в основном от вязкости битума поступающего в диспергатор вместе с раствором

Таблица 3.2: Распределение частиц на рис. 3.10

График	мин. размер частиц, мкм.	макс. размер частиц, мкм.	Среднее значение, мкм.	Ср.кв. от- клонение, мкм.	Всего ча- стиц
Исходный	0.87	2.98	1.71	0.44	100
Распознанный	0.87	2.52	1.6	0.46	57

ПАВ, а так же от скорости вращения и вида диспергирующих элементов. После выхода готовой эмульсии из диспергатора необходимо оценить её качество.

Внешние признаки распада эмульсии видны невооруженным глазом, но если речь идет о количественном сравнении двух визуально схожих эмульсий, то в таком случае применение программы для анализа размера частиц будет весьма кстати.

3.2.4 Определение среднего размера и дисперсии частиц битумной эмульсии на модифицированном битуме

Стабильность эмульсии в большой степени определяется размером частиц, который в свою очередь зависит от вязкости исходного битума. Во многих практических ситуациях необходимо получать мелкие (1-5 мкм) частицы эмульсии, это даёт очень хорошую стабильность при хранении и хорошее обволакивание заполнителей. Для получения таких эмульсий необходимо специализированное дорогостоящее оборудование.

Таблица 3.3: Распределение частиц на рис.3.11

Среднее значение, мкм.	Ср.кв. от- клонение, мкм.	Всего ча- стиц
12.86	5.06	108

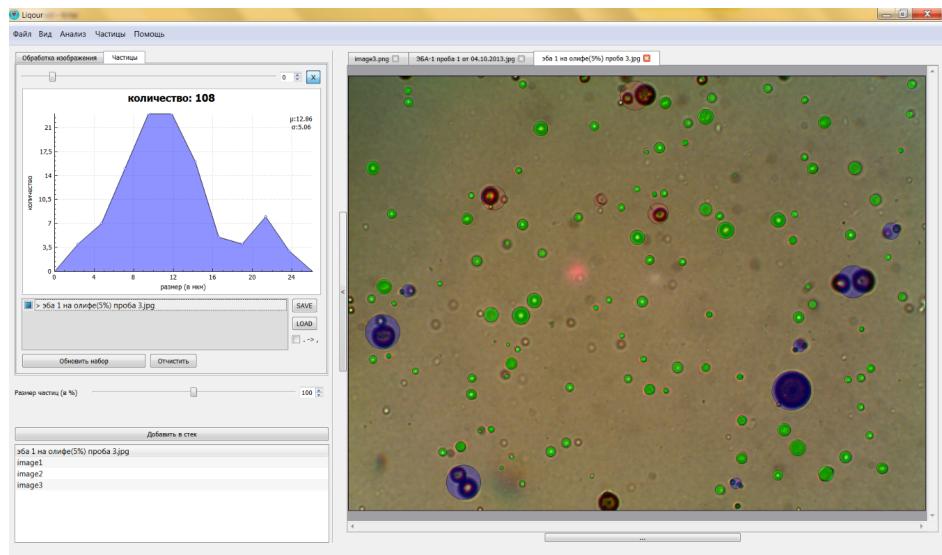


Рисунок 3.11: ЭБА-1 на битуме разжиженном олифой (5%)

Стандартные диспергаторы на которых дорожники производят битумные эмульсии позволяют получать средний размер частиц примерно 10 -20 мкм. Одним из возможных способов уменьшения размеров частиц эмульсии может являться понижение вязкости битума.

3.2.5 Результаты

Разработанный программный комплекс, показал что изложенные выше алгоритмы отлично справляются с задачей классификации объектов по контуру.

3.3 Автоматизация составления ПОДД

Работая с базой данных (БД) автомобильных дорог (далее а/д), было замечено, что большинство свойств а/д, можно отразить следующей схемой (рис. 3.12, серая область).

где состояние – это некоторый участок АД, а свойство - это функция, протяженная на этом участке и соответствующая некоторому свойству дороги (причем в качестве свойств может выступать наличие на участке трубы, дорожного знака, съезда и т.д.). Стоит отметить, что в БД функция представлена, как правило, набором аппроксимирующих точек.

В соответствии с таким подходом, было решено разработать надстройку над БД, представляющую собой набор фиксированных таблиц, и осуществляющую

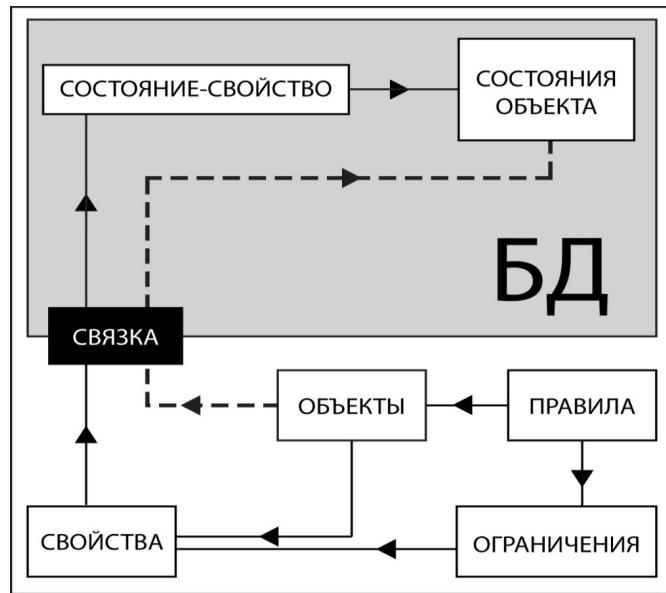


Рисунок 3.12: Схема представления свойств автомобильной дороги

связь с основной БД, через таблицу «связка». Таблица «связка» имеет ключевое значение в схеме. За счет неё организуется «безболезненная» связь между основной БД и надстройкой.

Таблица 3.4: Таблица связи

id	имя таблицы в связи	имя поля содержащего значение в таблице связи
##	состояние- свойство	значение

Так как схема разрабатывалась для общего случая автоматизация размещения подобъектов (т.е. в нашем случае дорожных ограждений, сигнальных устройств и т.д.) на протяженном объекте (а/д), то стоит разъяснить значения некоторых таблиц.

Таблица «объекты» хранит все возможные протяженные объекты, на которых будут размещаться подобъекты. В нашем случае, в таблице будет только одна строка, соответствующая АД. Таблица «состояния объекта» хранит всевозможные участки АД (т.е. например Иркутск-Листвянка, Братск-Усть-Илимск и т.д.).

Таблица 3.5: Таблица правил

id правила	id объекта	наименование	id списка ограничений
##	##	ГОСТ ####	##

Таблица «состояние-свойство» хранит множество функций-свойств (высота насыпи, тип покрытия и т.д.) для каждого конкретного глобального участка (состояния) АД. Об особенностях представления правил будет сказано ниже.

3.3.1 Представление дороги

Для представления а/д, заданной на промежутке, используется система функций:

$$Road = \begin{cases} g_1(x), x \in [a, b] \\ g_2(x), x \in [a, b] \\ \dots \\ g_n(x), x \in [a, b] \end{cases}$$

где $g_i(x)$ соответствует i -му свойству. Если свойство есть некоторая постоянно изменяющиеся величина (например «высота насыпи»), то функция имеет вид непрерывной кривой и при технической реализации представляет собой набор аппроксимирующих точек. Если же свойство представляет собой величину, которая может принимать значения только из строго заданного набора (например «тип покрытия»), то функция имеет ступенчатый вид.

Таблица 3.6: Таблица ограничений

id ограничения	id свойства	значение
##	##	асфальт

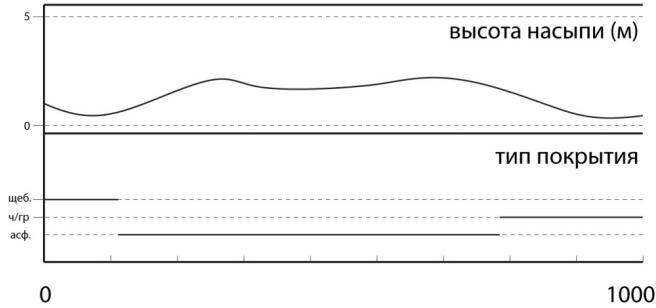


Рисунок 3.13: графики свойств

3.3.2 Представление правил

В отличие от данных а/д, информацию о правилах в БД хранить не принято. Связанно это с тем что, при составлении ПОДД, вся ответственность о корректности расположения тех или иных объектов на дороге ложится на плечи проектировщика, который прекрасно владеет правилами, и в случае необходимости обращается к ГОСТам. С другой стороны, чтобы научить машину расставлять объекты на плане в соответствии с ГОСТАми в автоматическом режиме, мы должны эти ГОСТы положить в БД.

Тут мы сталкиваемся с первой проблемой: с формальным представлением правила (т.е. как нам представить правило, чтобы его можно было хранить в БД). Так как каждое правило это есть некоторый набор конкретных значений свойств АД, то мы и будем представлять правило просто как набор значений:

$$P_j = (p_{j_1}, \dots, p_{j_n})$$

где каждая p_{j_i} есть константа и соответствует i -му свойству j -го правила. В более сложных случаях, например, когда значение свойства может принимать значения из некоторого заданного промежутка, можно заменить константы векторами:

$$P_j = (\overline{p_{j_1}}, \dots, \overline{p_{j_n}})$$

где $\overline{p_{j_i}} = (a_{j_i}, b_{j_i})$ и соответствует i -му свойству j -го правила. Оба варианта легко реализуются в рамках СУБД.

Вторая проблема заключается в том, как ГОСТ привести к формальному виду. Рассмотрим решение данной проблемы на примере ГОСТ 52289 пункта 8,

правила применения дорожных ограждений и направляющих устройств. Так как постановка задачи требует от нас найти корректное расположение для некоторого объекта на АД, то можно пренебречь некоторыми правилами определяющими качественное представление объекта (напр.: кол-во направляющих устройств, удерживающая способность и т.д.).

В качестве наглядного представления формализованного правила удобно использовать блок-схемы.

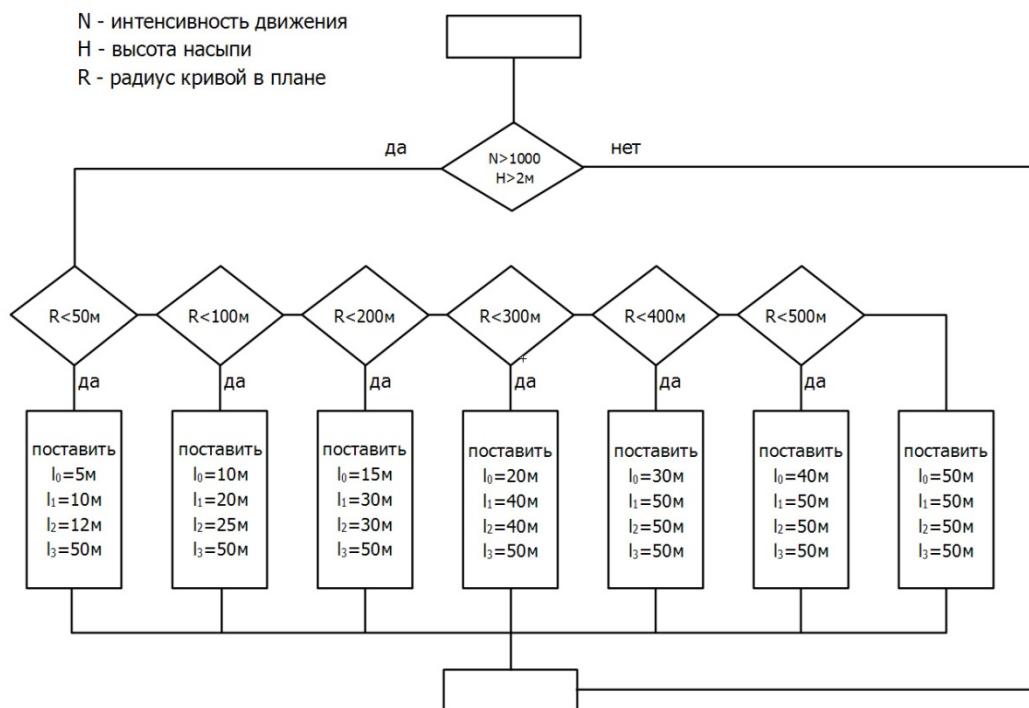


Рисунок 3.14: Блок схема определяющая ГОСТ 52289 пункта 8

На рис. 3.14. приведена часть схемы для расстановки сигнальных устройств на АД. Каждый возможный «путь» в этой схеме будет соответствовать одной строке в таблице правил БД.

3.3.3 Автоматизация

Разработанная система представления, позволяет нам реализовать схему автоматизированного расставления объектов на плане АД. Для начала рассмотрим простой пример, который позволит нам лучше разъяснить принцип работы. Пусть АД определена в базе двумя функциями-свойствами *A* и *B* и задано два правила:

$$P_1 = [(a_1, a_2), b_1]$$

$$P_2 = [(a_2, a_3), b_2]$$

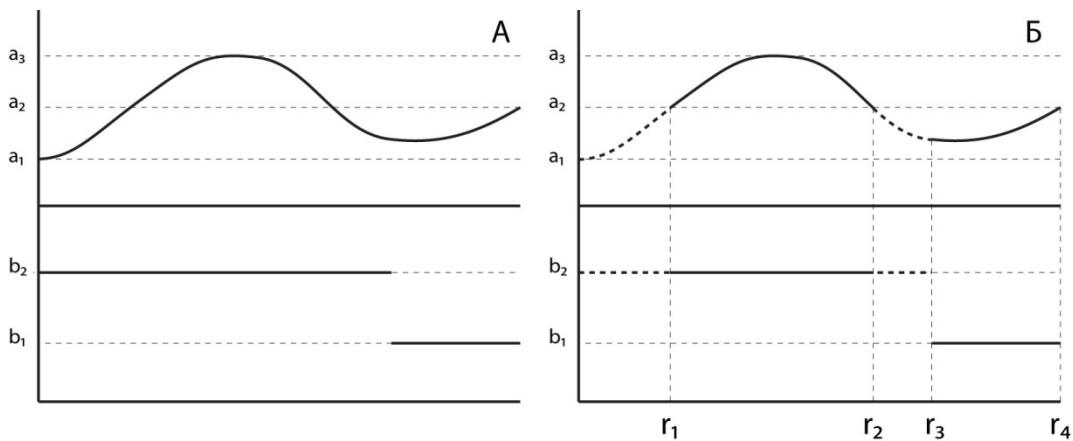


Рисунок 3.15: Объединенные графики свойств

На рис. 3.15 А представлены графики функций-свойств A (верхний) и B (нижний) на некотором участке дороги. Наша задача найти промежутки удовлетворяющие заданным выше правилам. Мы разбиваем участок на под-участки точками пересечения функций $A(x)$ и $B(x)$ с прямыми $y = a_1, y = a_2, y = a_3$ и $y = b_1, y = b_2$ соответственно.

В результате обнаруживаем (рис. 3.15 Б), что промежуток $[r_1, r_2]$ удовлетворяет правилу P_2 , а промежуток $[r_3, r_4]$ удовлетворяет правилу P_1 . После того как промежутки найдены, остается подобрать объекты для которых выполнения любого из правил P_1 и P_2 достаточно для того, чтобы их расположение соответствовало правилам ГОСТ, и поставить их на плане АД.

Формально автоматизацию можно описать следующим образом. Дорога представлена системой функций

$$Obj = \begin{cases} g_1(x), x \in [a, b] \\ g_2(x), x \in [a, b] \\ \dots \\ g_n(x), x \in [a, b] \end{cases}$$

заданной в двумерном пространстве на промежутке $[a, b]$. Функции $g_i(x)$ будем называть свойствами объекта. Функции $g_i(x)$ являются непрерывными по X на промежутке $[a, b]$. Требуется найти промежутки из $[a, b]$, на которых объект удо-

вляет ограничениям. Пусть A_i как области значений функций определяющих ограничение:

$$A_i = E(f_i(x)), i = \overline{1, n}$$

где $f_i(x)$ – функции заданные на некотором промежутке. Ограничения будем представлять следующим образом:

$$R_j = \begin{cases} f_1(x_j) \\ f_2(x_j) \\ \dots \\ f_n(x_j) \end{cases}, j = \overline{1, m}$$

Этап 1. Сведение объекта к новому объекту заданному ступенчатыми функциями.

$$Obj_2 = \begin{cases} h_1(x), x \in [a, b] \\ h_2(x), x \in [a, b] \\ \dots \\ h_n(x), x \in [a, b] \end{cases} \quad h_i = \{x \mid f_i(x_{j_1}) \leq g_i(x) < f_i(x_{j_2})\}$$

Оценка: сложность осуществления перехода порядка $O(m)$.

Этап 2. Разбиение объекта на множество подобъектов $\{obj_k\}$. Данный шаг позволяет нам перейти от объекта заданного системой функций к множеству подобъектов, каждый из которых определен системой констант. Такое разбиение позволяет нам реализовать методы проверки ограничений, который подробно были рассмотрены в предыдущих работах [3].

Пусть s_i – кол-во точек разрыва у функции $h_i(x)$ на $[a, b]$. $S = \sum_i s_i$ – это общее кол-во точек разрыва на $[a, b]$ у всех функций $h_i(x)$ вместе взятых.

Пусть $\{x_1, \dots, x_S\}$ – упорядоченное по возрастанию множество точек разрыва. Тогда наше множество подобъектов будет задано следующим образом:

$$obj_i = \begin{cases} h_1^+(x_i), x \in [x_i, x_{i+1}] \\ h_2^+(x_i), x \in [x_i, x_{i+1}] \\ \dots \\ h_n^+(x_i), x \in [x_i, x_{i+1}] \end{cases} \quad \text{при } i < S - 1$$

и

$$obj_i = \begin{cases} h_1^+(x_i), x \in [x_i, b] \\ h_2^+(x_i), x \in [x_i, b] \\ \dots \\ h_n^+(x_i), x \in [x_i, b] \end{cases} \quad \text{при } i = S - 1$$

Отметим что $x_1 = a, x_s = b$. Сложность построения нового множества объектов порядка $O(S)$.

Оценки позволяют сделать вывод о возможности эффективной реализации предложенного метода. Результаты Система автоматизации проектирования ПОДД успешно использовалась при выполнении контрактов составления ПОДД для а/д «Иркутск-Листвянка», «Братск-Усть-Илимск» и других.

Заключение

Основные результаты работы заключаются в следующем. На качественном уровне: для данных, организованных более сложно, чем таблицы реляционных баз данных, доказаны очень близкие результаты по вычислительной сложности их анализа. Более точно, основные результаты работы могут быть сформулированы следующим образом.

1. На основе анализа плоских контурных изображений разработана формализация, представленная ориентированными дугами, связями дуг и их численными характеристиками в градусном измерении и относительными размерами длины дуг.
2. Исследования показали, что вычислительная сложность анализа плоских контурных изображений, представленных ориентированными дугами, связями дуг и их численными характеристиками в градусном измерении и относительными размерами длины дуг, почти не зависит от количества образцов.
3. Для подтверждения теоретических результатов на практике было созданы комплексы программ для решения задач.

Полученные в диссертационной работе результаты соответствуют общей тенденции развития математических методов решения информационных задач для больших массивов данных.

Список рисунков

2.1 Сравнение методов обесцвечивания	25
2.2 Распространение волны: а)–ромбовидная, б)–квадратная, в)–сферическая	30
2.3 Распространение волны по области, на последнем изображении представлен полученный график	33
2.4 Схема разбиения графа «особыми» точками	34
3.1 Графический редактор	60
3.2 Разный шаг аппроксимации графа	60
3.3 Преобразованное изображение	61
3.4 Обучение системы	61
3.5 Преобразованное изображение	62
3.6 Интерфейс интерпретатора	63
3.7 Скриншоты программы «анализ изображений»	66
3.8 Концентрированная битумная анионная эмульсия в отраженном свете	66
3.9 Результат работы программы на искусственно сгенерированных данных	67
3.10 Проверка работы программы при наличии слипшихся частиц	68
3.11 ЭБА-1 на битуме разжиженном олифой (5%)	69
3.12 Схема представления свойств автомобильной дороги	70
3.13 графики свойств	73
3.14 Блок схема определяющая ГОСТ 52289 пункта 8	74
3.15 Объединенные графики свойств	75

Список таблиц

3.1	Анализ распределения частиц на рис. 3.9	67
3.2	Распределение частиц на рис. 3.10	68
3.3	Распределение частиц на рис.3.11	70
3.4	Таблица связки	71
3.5	Таблица правил	72
3.6	Таблица ограничений	72

Литература

1. Big Data Computing and Clouds: Challenges, Solutions, and Future Directions / Marcos D. Assuncao, Rodrigo N. Calheiros, Silvia Bianchi [и др.] // Technical Report CLOUDS-TR-2013-1, Cloud Computing and Distributed Systems Laboratory, The University of Melbourne. 2013.
2. В.И. Мартьянов. Логико-эвристические методы сетевого планирования и распознавание ситуаций // Труды Международ. конф. «Проблемы управления и моделирования в сложных системах». 2001. С. 203–215.
3. Bigtable: A Distributed Storage System for Structured Data / Fay Chang, Jeffrey Dean, Sanjay Ghemawat [и др.]. 2006.
4. А.И. Мальцев. Алгебраические системы. М.: Наука., 1967.
5. С.В. Яблонский. Введение в дискретную математику. М.:Наука, 1979. С. 272.
6. А.И. Кокорин, А.Г. Пинус. Вопросы разрешимости расширенных теорий // УМН. 1978. Т. 2. С. 49–84.
7. В. Манцивода А. О р-преобразованиях формул // Тез. докл. 7-й Всесоюз. конф. по мат. логике.– Новосибирск : ИМ СО АН СССР. 1984.
8. E.F. Codd. A Relational Model of Data for Large Shared Data Banks // CACM. 1970.
9. E.F. Codd. The Relational Model For Database Management Version 2. Reading, Mass.: Addison-Wesley, 1990.
10. Д. Кнут. Искусство программирования для ЭВМ. Сортировка и поиск. М.: Мир, 1978. С. 848.

11. Обзор приложений логико-эвристических методов решения комбинаторных задач / Мартынов В.И., Архипов В.В., Каташевцев М.Д. [и др.] // Материалы 3-ей Российской школы – семинара «Синтаксис и семантика логических систем». - Иркутск. 2010. С. 60–64.
12. Обзор приложений логико-эвристических методов решения комбинаторных задач высокой сложности / Мартынов В.И., Архипов В.В., Каташевцев М.Д. [и др.] // Современные технологии. Системный анализ. Моделирование. ИрГУПС. 2010. С. 61–67.
13. Р. Беллман. Динамическое программирование. М.: Изд-во иностранной литературы, 1960.
14. М. Гери, Д. Джонсон. Вычислительные машины и труднорешаемые задачи. М.: Мир, 1982.
15. Д.М. Еремин, И.Б. Гарцеев. Искусственные нейронные сети в интеллектуальных системах управления. М.: МИРЭА, 2004.
16. Ж.-Л. Лорье. Системы искусственного интеллекта. М.: Мир, 1991.
17. М.Д Каташевцев, В.И. Мартынов. Логико-эвристические методы анализа плоских изображений // Известия Иркутского университета: ежегод.науч.-теорет. конф. аспирантов и студентов: материалы. – Иркутск: Изд-во Иркут. гос.ун-та,. 2010. С. 175–177.
18. М.Д. Каташевцев. Волновая скелетизация // Вестник Иркутского Государственного Технического Университета. 2013. С. 89–92.
19. Автоматизированная технология создания проектов организации дорожного движения / Каташевцев М.Д., Мартынов В.И., Степаненко А.А. [и др.] // Вестник ИрГТУ. 2012. С. 150–155.
20. М.Д. Каташевцев. Анализ плоских контурных изображений с метрикой // Известия Иркутского Государственного Университета серия «Математика». 2014. С. 39–48.

21. М.Д Каташевцев, В.И. Мартынов. Программа обработки и интерпретации контурных изображений // Свидетельство о государственной регистрации программы для ЭВМ №2011618417.- Федеральная служба по интеллектуальной собственности, патентам и товарным знакам.- Москва. 2010.
22. И. Мартынов В., М.Д. Каташевцев. Комбинаторные задачи высокой сложности и анализ плоских контурных изображений // Известия Иркутского Государственного Университета серия «Математика». 2013. С. 31–47.
23. Автоматизация создания проектов организации дорожного движения для автомобильных дорог / Пахомов Д.В., Каташевцев М.Д., Мартынов В.И. [и др.] // Современные технологии. Системный анализ. Моделирование. ИрГУПС. 2010. С. 56–61.

Приложение А

Структура БД

ОБРАЗЦЫ

Таблица **Smpl_Arc**, задающая совокупность дуг образца

1. id идентификатор дуги (int 4);
2. min_sector минимально возможный угол сектора дуги (int 4);
3. max_sector максимально возможный угол сектора дуги (int 4);
4. clockwise направление обхода (1 – по солнцу; -1 – против солнца; 0 – неопределенno) (int 4);
5. bran_beg кол-во ветвлений на начале
6. bran_end кол-во ветвлений на конце
7. quan_circle кол-во циклов, включающих дугу

Таблица **Smpl_Arcs_Lists**, задающая идентификаторы списков дуг образца

1. id идентификатор списка дуг (int 4);

Таблица **Smpl_Arcs_Lists_Arcs**, задающая списки дуг образца

1. id_arcslist идентификатор списка дуг (int 4);
2. id_arc идентификатор дуги (int 4);
3. proportion относительная величина дуги

Таблица **Smpl_Relations**, задающая совокупность связей дуг образца

1. id идентификатор связи двух дуг (int 4);

2. id_arc1 идентификатор первой дуги (int 4);
3. id_arc2 идентификатор второй дуги (int 4);
4. min_angle минимально возможный угол пересечения дуг (int 4);
5. max_angle максимально возможный угол пересечения дуг (int 4);
6. type тип связи (0: конец – начало, 1: конец – конец, 2: начало - начало)

Таблица Smpl_Relations_Lists, задающая идентификаторы списков связей дуг образца

1. id идентификатор списка связей дуг (int 4);

Таблица Smpl_Relations_Lists_Relations, задающая списки связей дуг образца

1. id_relationslist идентификатор списка связей дуг (int 4);
2. id_relation идентификатор связи двух дуг (int 4);

Таблица Smpl_Parts, задающая части образца

1. id идентификатор части образца (int 4);
2. id_arcslist идентификатор списка дуг для исследуемого изображения (int 4);
3. id_relationslist идентификатор списка связей дуг для исследуемого изображения (int 4);
4. hor_angle угол к горизонту для первой дуги в списке id_arcslist (int 4);
5. proportion пропорция (float 8)

Таблица Smpl_Parts_Relations, задающая связи частей образца

1. id идентификатор символа (int 4);
2. id_part1 идентификатор части образца (int 4);
3. id_part2 идентификатор части образца (int 4);
4. min_pos_angle (int 4);
5. max_pos_angle (int 4);
6. min_central_angle (int 4); не использовать пока
7. max_central_angle (int 4); не использовать пока
8. type тип связи(0 – снаружи; 1 – внутри) (int 4);

Таблица Smpl_Parts_Lists, задающая идентификаторы списков частей образца

1. id идентификатор списка частей образца (int 4);

Таблица Smpl_Parts_Lists_Parts, задающая списки частей образца

1. id_partslist идентификатор списка частей образца (int 4);
2. id_part идентификатор части образца (int 4);

Таблица Smpl_Parts_Relations_Lists, задающая идентификаторы списков связей частей образца

1. id идентификатор символа (int 4);

Таблица Smpl_Parts_Relations_Lists_Parts_Relations, задающая списки связей частей образца

1. id_partsrelationslist идентификатор списка частей образца (int 4);
2. id_partsrelation идентификатор части образца (int 4);

Таблица Smpl_Samples, задающая совокупность образцов

1. id идентификатор образца (int 4);
2. id_partslist идентификатор списка частей для исследуемого изображения (int 4);
3. id_partsrelationslist идентификатор списка связей частей для исследуемого изображения (int 4);
4. presentation графическое описание образца (image 16);
5. description описание образца (char 250);

СИМВОЛЫ

Таблица Smbl_Symbols, задающая совокупность символов

1. id идентификатор символа (int 4);
2. symbol имя символа (char 250);
3. description описание символа (char 250);
4. presentation графическое описание символа (image 16);
5. id_sampleslist идентификатор списка образцов, представляющих образец (int 4);

Таблица Smbl_Samples_Lists, задающая идентификаторы списков образцов

1. id идентификатор списка символов (int 4);

Таблица Smbl_Samples_Lists_Samples, задающая списки образцов

1. id_sampleslist идентификатор списка образцов (int 4);
2. id_sample идентификатор образца (int 4);

ИЗОБРАЖЕНИЯ

Таблица Img_Arc, задающая совокупность дуг изображения

1. id идентификатор дуги (int 4);
2. min_sector минимально возможный угол сектора дуги (int 4);
3. max_sector максимально возможный угол сектора дуги (int 4);
4. clockwise направление обхода (1 – по солнцу; -1 – против солнца; 0 – неопределенno) (int 4);
5. bran_beg кол-во ветвлений на начале
6. bran_end кол-во ветвлений на конце
7. quan_circle кол-во циклов, включающих дугу

Таблица Img_Arcs_Lists, задающая идентификаторы списков дуг изображения

1. id идентификатор списка дуг (int 4);

Таблица Img_Arcs_Lists_Arcs, задающая списки дуг изображения

1. id_arcslist идентификатор списка дуг (int 4);
2. id_arc идентификатор дуги (int 4);
3. proportion относительная величина дуги

Таблица Img_Relations, задающая совокупность связей дуг изображения

1. id идентификатор связи двух дуг (int 4);
2. id_arc1 идентификатор первой дуги (int 4);
3. id_arc2 идентификатор второй дуги (int 4);
4. min_angle минимально возможный угол пересечения дуг (int 4);
5. max_angle максимально возможный угол пересечения дуг (int 4);
6. type тип связи (0: конец – начало, 1: конец – конец, 2: начало – начало)

Таблица Img_Relations_Lists, задающая идентификаторы списков связей дуг изображения

1. id идентификатор списка связей дуг (int 4);

Таблица Img_Relations_Lists_Relations, задающая списки связей дуг изображения

1. id_relationslist идентификатор списка связей дуг (int 4);
2. id_relation идентификатор связи двух дуг (int 4);

Таблица Img_Parts, задающая части изображения

1. id идентификатор части изображения (int 4);
2. id_arcslist идентификатор списка дуг для исследуемого изображения (int 4);
3. id_relationslist идентификатор списка связей дуг для исследуемого изображения (int 4);
4. hor_angle угол к горизонту для первой дуги в списке id_arcslist (int 4);
5. proportion пропорция (float 8)

Таблица Img_Parts_Lists, задающая идентификаторы списков частей изображения

1. id идентификатор списка частей образца (int 4);

Таблица Img_Parts_Lists_Parts, задающая списки частей изображения

1. id_partslist идентификатор списка частей образца (int 4);
2. id_part идентификатор части образца (int 4);

Таблица Img_Parts_Relations, задающая связи частей изображения

1. id идентификатор символа (int 4);
2. id_part1 идентификатор части образца (int 4);
3. id_part2 идентификатор части образца (int 4);
4. min_pos_angle (int 4);
5. max_pos_angle (int 4);
6. min_central_angle (int 4); не использовать пока
7. max_central_angle (int 4); не использовать пока
8. type тип связи(0 – снаружи; 1 – внутри) (int 4);

Таблица Img_Parts_Relations_Lists, задающая идентификаторы списков связей частей изображения

1. id идентификатор символа (int 4);

Таблица Img_Parts_Relations_Lists_Parts_Relations, задающая списки связей частей изображения

1. `id_partsrelationslist` идентификатор списка частей образца (int 4);
2. `id_partsrelation` идентификатор части образца (int 4);

Таблица `Img_Samples`, задающая совокупность изображений

1. `id` идентификатор образца (int 4);
2. `id_partslist` идентификатор списка частей для исследуемого изображения (int 4);
3. `id_partsrelationslist` идентификатор списка связей частей для исследуемого изображения (int 4);
4. `presentation` графическое описание образца (image 16);
5. `description` описание образца (char 250);