

МИЭТ

Работа допущена к защите
зав. кафедрой

Прокофьев Александр Александрович

«_____» _____ 2018 г.

ВЫПУСКНАЯ РАБОТА БАКАЛАВРА

Тема: **Упрощение топологии объектов на цифровых
географических картах**

Направление: 01.03.04 – Прикладная математика

Выполнил студент гр. МП-40

_____ Березин Александр Андреевич

Научный руководитель,

ст. преподаватель

_____ Назаров Максим Николаевич

Москва – 2018

Оглавление

1.	Введение	3
2.	Постановка задачи	3
3.	Обзор предметной области	4
3.1.	Стандарт OpenGIS	4
3.2.	Существующие решения	7
3.3.	Теоретические сведения	7
4.	Общая методология	10
4.1.	Требование 1	10
4.2.	Требование 2	12
4.3.	Требование 3	13
4.4.	Требование 4	15
4.5.	Проверка допустимости соединений	15
5.	Реализация алгоритма	17
5.1.	Интерфейс	18
5.2.	Объектная модель	20
5.3.	Пример работы	21
6.	Заключение	21
	Список литературы	24

1. Введение

В работе описывается решение задачи преобразования многосвязных геометрических областей в односвязные в контексте кодирования географических карт. Полученный алгоритм работает в общем случае и может использоваться в других предметных областях.

Речь, конечно, идет о схематических картах, а не о спутниковых фотографиях. Такие карты обычно кодируются в векторном формате. Но конкретная форма представления данных может отличаться в различных геоинформационных системах.

По условию поставленной задачи имеются две геоинформационные системы, которые используют различные представления данных. Первая поддерживает в составе карты объекты любого размера и топологии. Вторая поддерживает только односвязные, а также имеет ограничение на количество точек в одном объекте. Понятно, что первая система автоматически поймет данные второй, но обратное неверно. Задача состоит в том, чтобы разработать алгоритм для преобразования данных из первой системы во вторую. Также на него накладываются дополнительные условия – создавать новые вершины запрещено, а вносимые разрезы должны быть минимальны.

2. Постановка задачи

В качестве входных данных имеется географическая карта, закодированная по стандарту Geography Markup Language (GML). Этот стандарт предназначен для кодирования областей различной природы – к примеру, лесных массивов, водоемов или островов – в виде вложенных многоугольников при помощи XML-грамматики [1].

Требуется преобразовать входные данные в новый GML-файл таким образом, чтобы одновременно выполнялись следующие условия:

1. Отсутствуют вложенные области, т.е. все области односвязны.
2. Число точек, задающих границу каждой области, не превышает заданной константы N.
3. Количество добавочных соединений и их длины минимальны.
4. Создание новых вершин запрещено.

Возможность работы в реальном времени не требуется, поэтому оптимизация алгоритма в данной работе не затрагивается.

3. Обзор предметной области

3.1. Стандарт OpenGIS

GML (Geography Markup Language) является спецификацией OpenGIS® Implementation, в которой определяется кодировка XML для передачи и хранения географической информации. Спецификация опубликована на веб-сайте Открытого геопространственного консорциума (OGS).

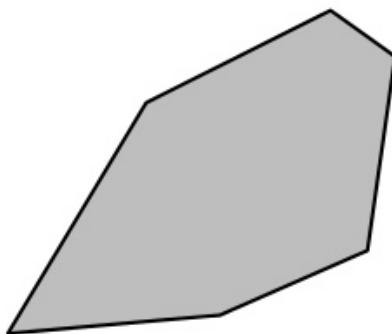


Рис. 1. Пример кодирования многоугольника при помощи GML-разметки

```
<gml:Polygon gml:id="p3" srsName="urn:ogc:def:crs:EPSG
:6.6:4326">
<gml:exterior>
```

```

<gml:LinearRing>
  <gml:coordinates>5,5 28,7 44,14 47,35 40,40 20,30
    5,5</gml:coordinates>
</gml:LinearRing>
</gml:exterior>
</gml:Polygon>

```

Open Geospatial Consortium (OGC) — международная некоммерческая организация, ведущая деятельность по разработке стандартов в сфере геопространственных данных и сервисов, созданная в 1994 году. В настоящее время координирует деятельность более 500 [2] правительственныех, коммерческих, некоммерческих и научно-исследовательских организаций с целью разработки и внедрения консенсусных решений в области открытых стандартов для геопространственных данных, обработки данных геоинформационных систем и совместного использования данных. Большинство стандартов OGC основано на принципах, изложенных в базовой модели данных для представления географических характеристик под названием Abstract Specification. На основе базовой модели участники консорциума разработали и продолжают разрабатывать большое число спецификаций или стандартов для обслуживания конкретных потребностей организаций-участников в области геопространственных технологий и сервисов, включая ГИС.

Проект OpenGIS был инициирован до учреждения OGS, и впоследствии стал торговым знаком для ряда спецификаций, разрабатываемых и поддерживаемых в рамках этого консорциума. Стандарт GML определяет спецификацию географического языка разметки на основе синтаксиса XML. Этот язык позволяет кодировать географическую информацию, которая включает, в частности, геометрические характеристики объектов, в форме XML-документов. Такие документы могут использоваться для обмена геоданными между различными приложениями, обеспечивая их унификацию; для хранения геоданных и доступа к

ним, в том числе и в среде глобальной сети; для публикации их на веб-сайтах. Разработка начальной версии языка GML (GML 1.0) была завершена весной 2000 года. Действующая в настоящее время версия этого стандарта - GML 3.3. Версия GML 3.1.0 была представлена консорциумом OGC в ISO в 1994 г. для создания на ее основе официального международного стандарта, где в настоящее время эта спецификация проходит принятую в ISO процедуру стандартизации [3].

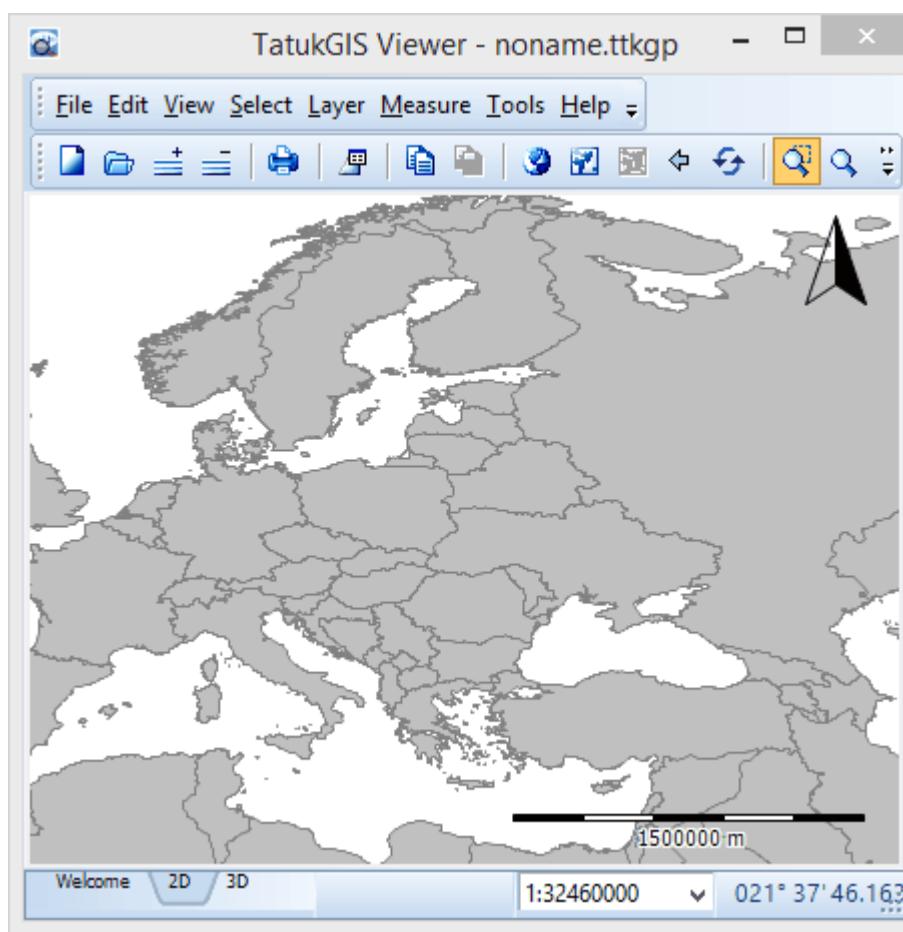


Рис. 2. Интерфейс программы TatuGIS Viewer

Для работы с данными, закодированными в GML и сходных стандартах, компанией TatuGIS была разработана соответствующая линейка приложений. В частности, для визуализации данных в данной работе использовался TatuGIS Viewer.

3.2. Существующие решения

Большинство алгоритмов упрощения топологии ГИС-файлов предназначаются для количественных преобразований (уменьшение размера файла), а не качественных. Однако, подобные задачи возникают в теории дифференциальных уравнений в частных производных и теории функций комплексной переменной — в частности, при доказательстве интегральной теоремы Коши для многосвязной области [4]; а также в теории дифференциальных уравнений в частных производных — например, при решении уравнения Лапласа в многосвязной области [5].

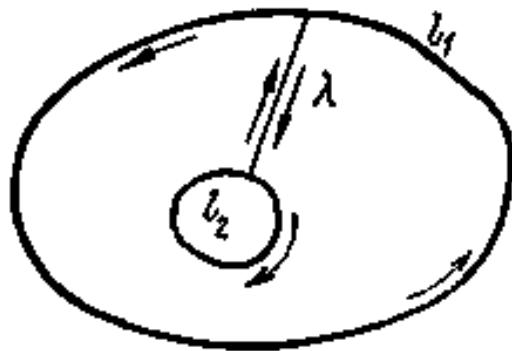


Рис. 3. Интегральная теорема Коши для многосвязной области

Задача 4.2, а, точнее, ее предельный случай при $N = 3$ (задача о триангуляции многоугольника [6]), широко известна в области трехмерной графики, поскольку в ней треугольник является фундаментальной единицей для составления более сложных объектов. Однако, прямое использование алгоритма триангуляции для решения задачи 4.2 очень неэффективно.

3.3. Теоретические сведения

Топология

Топология — раздел математики, занимающийся изучением свойств фигур (или пространств), которые сохраняются при непрерывных деформациях, таких, например, как растяжение, сжатие или изгибание. Непрерывная деформация — это деформация фигуры, при которой не происходит разрывов (т.е.

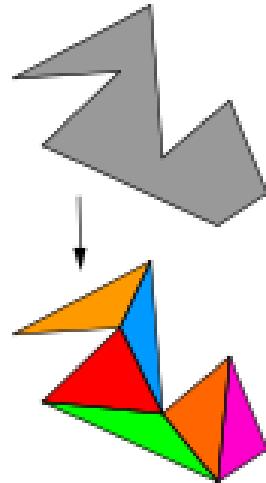


Рис. 4. Задача о триангуляции многоугольника

нарушения целостности фигуры) или склеиваний (т.е. отождествления ее точек). Такие геометрические свойства связаны с положением, а не с формой или величиной фигуры.

Одной из характеристик фигур, инвариантных относительно непрерывных деформаций, является порядок связности. В 4.1 использовалось понятие односвязности. В простейшем двумерном случае односвязная фигура — такая, в которой любой замкнутый путь можно непрерывно стянуть в точку. Проще говоря, это фигура, в которой отсутствуют «дыры». Двусвязная область имеет одну дыру, трехсвязная — две, и т.д. При этом форма областей и дыр не имеет значения [7].

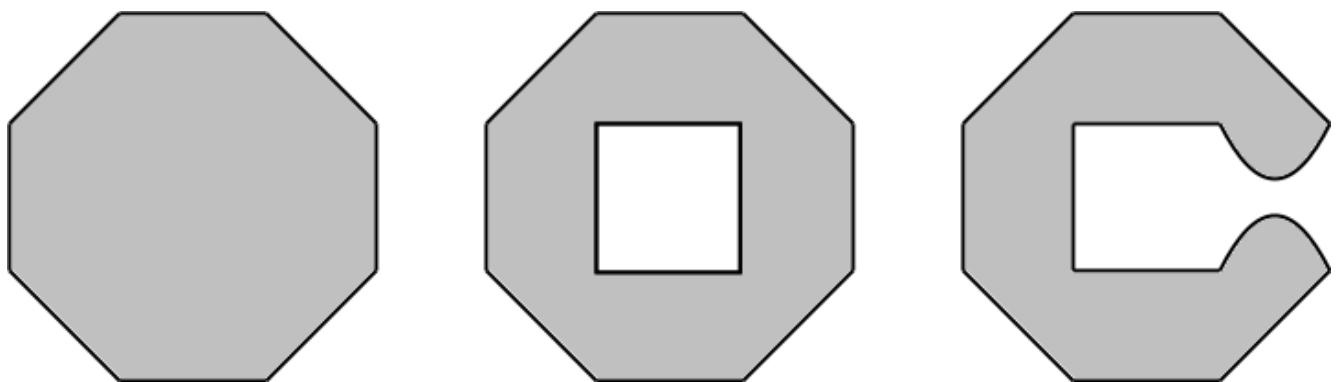


Рис. 5. Слева направо: односвязная область, двусвязная область, преобразование двусвязной области в односвязную

Поскольку поставленная задача состоит в изменении порядка связности,

конкретно — в его уменьшении, она не может быть решена при помощи непрерывных деформаций. Требуется внести в многосвязные области разрезы при минимальном изменении их существенных характеристик. Эта цель достигается при помощи вырожденных туннелей: отрезок, по которому производится разрез, дважды вносится в границу результирующей области, благодаря чему площадь области не изменяется.

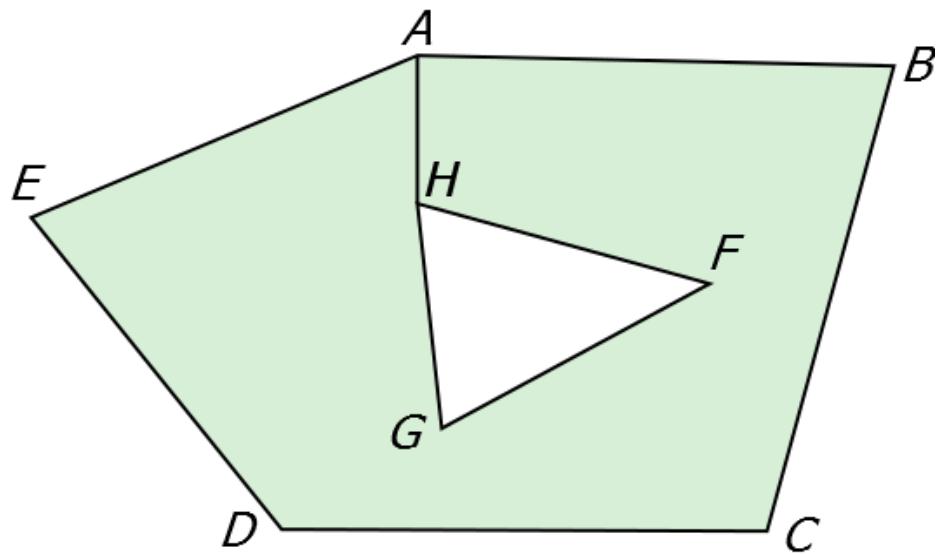


Рис. 6. Упрощение топологии полигональной области: $ABCDE \setminus FGH = ABCDEAHGFHAB$

Теория графов

Для решения задачи 4.1 будет использован *алгоритм Прима* из теории графов.

Граф — это совокупность двух множеств: множества точек, которые называются вершинами, и множества ребер А. Ребра — это неупорядоченные (для неориентированного графа) или упорядоченные (для ориентированного) пары элементов из множества вершин, которые называются концами ребра. Если каждому ребру сопоставляется некоторое число (вес), то граф называетсязвешенным.

Дерево — это граф, не имеющий циклов, т.е. замкнутых путей. Остов графа — это его подграф, содержащий все вершины исходного, но являющийся

деревом. Минимальный остов взвешенного графа — это оставное дерево, имеющее наименьший суммарный вес ребер [8].

Алгоритм Прима используется для поиска в неориентированном взвешенном графе минимального оставного дерева. В нем искомый минимальный остов строится постепенно, путем добавлением в него ребер по одному. Изначально остав полагается состоящим из единственной вершины, взятой произвольно. Затем выбирается ребро минимального веса, исходящее из этой вершины, и добавляется в минимальный остав. Далее на каждом шаге ищется минимальное по весу ребро, один конец которого — уже взятая вершина, а другой — ещё не взятая, и это ребро добавляется в остав (если таких рёбер несколько, можно взять любое). Этот процесс повторяется до тех пор, пока остав не будет содержать все вершины [9]. Без дополнительных оптимизаций алгоритм Прима работает за квадратичное время.

Альтернативой алгоритма Прима является алгоритм Краскала. В нем текущее множество ребер вначале устанавливается пустым. Затем, пока это возможно, проводится следующая операция: из всех рёбер, добавление которых к уже имеющемуся множеству не вызовет появление в нём цикла, выбирается ребро минимального веса и добавляется к уже имеющемуся множеству. Когда таких рёбер больше нет, алгоритм завершён. Подграф данного графа, содержащий все его вершины и найденное множество рёбер, является его оставным деревом минимального веса [10]. Время выполнения алгоритмов Прима и Краскала различается незначительно. В данной работе алгоритм Прима был выбран благодаря простоте реализации.

4. Общая методология

4.1. Требование 1

Формулировка требования: «Отсутствуют вложенные области, т.е. все области односвязны».

Многосвязная область состоит из внешней границы и произвольного числа внутренних границ, заданных многоугольниками. Задача снижения порядка связности области сводится к выбору соединений точек различных границ таким образом, чтобы все они оказались связаны. В простейшем случае с одной внутренней границей подойдет любой отрезок, один из концов которого принадлежит множеству вершин внешней границы, а другой — внутренней, и который не имеет с границами других общих точек. В случае трехсвязной области (с двумя внутренними границами) можно также соединить внутренние границы, а затем одну из них — со внешней.

Чтобы найти требуемые соединения, преобразуем объектную модель карты следующим образом. Поставим в соответствие каждому многоугольнику, включая внешнюю границу, вершину графа и сделаем полученный граф полным. В полученной графовой модели ребра соответствуют соединениям между многоугольниками. В качестве весов присвоим им длины кратчайших соединений между соответствующими многоугольниками. Далее, нужно исключить из модели недопустимые соединения — такие, которые имеют более двух общих точек с многоугольниками. Для этого будем проверять соединение на пересечение со всеми сторонами многоугольников в рассматриваемой области. Наконец, к полученному графу применим алгоритм Прима [9] для выделения минимального оствового дерева.

Восстановление объектной модели из графовой происходит следующим образом. На внешней границе произвольно выбирается начальная точка. Если с ней не связано ни одно из соединений в графовой модели, происходит переход к следующей вершине в том же многоугольнике; в противном случае эта вершина сохраняется в памяти, а следующей выбирается точка, находящаяся на другом конце найденного соединения, для которой та же процедура повторяется рекурсивно. Когда процесс возвращается в исходную точку, он возвращается по тому же соединению к предыдущему многоугольнику. Алгоритм завершает работу, когда возвращаться становится некуда. В результате вместо многосвязной обла-

Объектная модель

Графовая модель

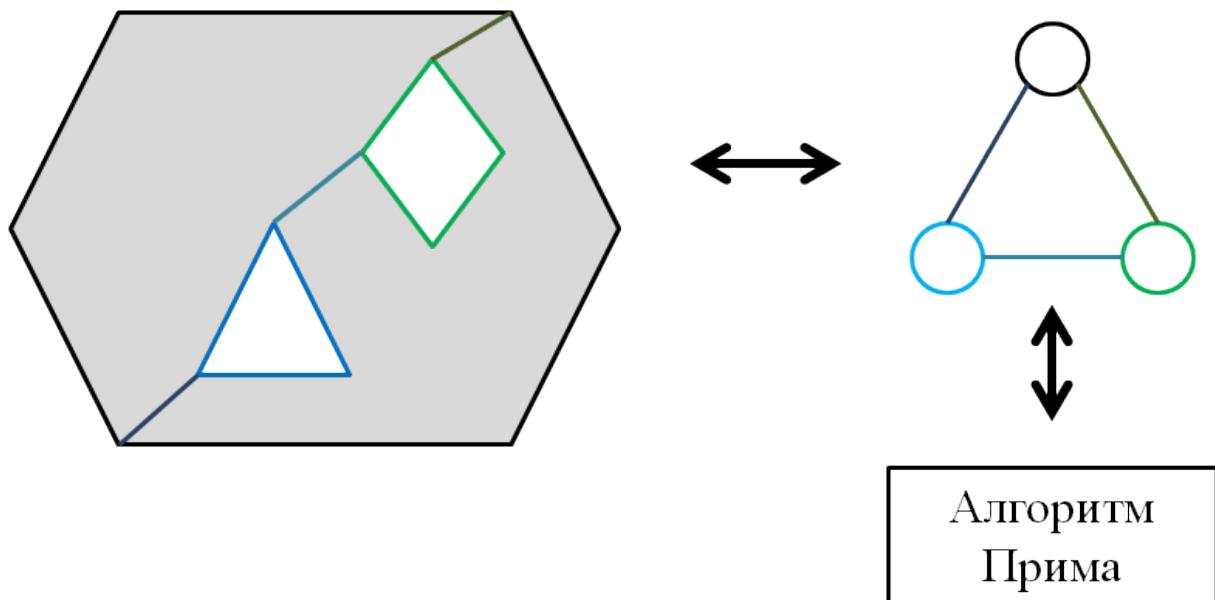


Рис. 7. Использование алгоритма Прима для удовлетворения требованию 1

сти получается односвязная, в которой соединения становятся вырожденными туннелями.

4.2. Требование 2

Формулировка требования: «Число точек, задающих границу каждой области, не превышает заданной константы N ». На данном этапе все области уже односвязны.

Рассмотрим вначале область, задаваемую выпуклым многоугольником с более чем N вершинами. Из определения выпуклости очевидно, что ее можно разделить в произвольном отношении, т.к. любая линия, соединяющая несмежные точки, будет лежать внутри области. С учетом требования 3 следует выбрать кратчайшую линию, разделяющую многоугольник таким образом, чтобы в одной из полученных частей было ровно N вершин. Если во второй части их больше N , повторяем ту же процедуру для нее рекурсивно.

Отметим, что при разбиении выпуклого многоугольника получаются так-

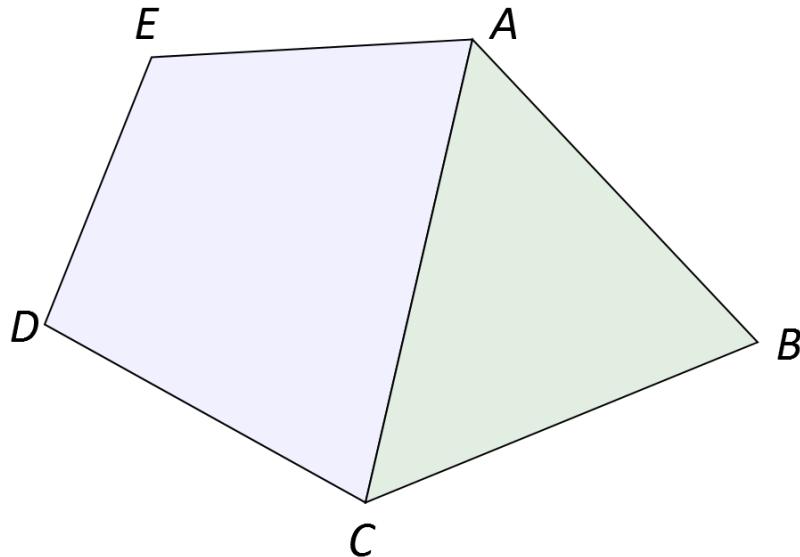


Рис. 8. Пример разбиения пятиугольника при максимальном количестве вершин $N=4$:
 $ABCDE = ABC \cup ACDE$

же выпуклые многоугольники, поскольку все вершины полученных многоугольников лежат по одну и ту же сторону от новой грани.

Рассмотрим теперь невыпуклые многоугольники. Согласно [6], любой многоугольник можно разбить на смежные треугольники. Очевидно, эти треугольники можно объединять по смежным сторонам, получая таким образом разбивку на части с произвольным количеством вершин. Следовательно, существует и такая разбивка, при которой одна из частей имеет ровно N вершин. Значит, описанный алгоритм можно распространить и на невыпуклые области.

4.3. Требование 3

Формулировка требования: «Количество добавочных соединений и их длины минимальны». В процессе разработки к этому условию было добавлено важное уточнение: если минимизация обоих параметров одновременно невозможна или невыполнима за полиномиальное время, приоритет имеет количество соединений.

Докажем, что после выполнения описанных выше шагов алгоритма это условие уже выполнено.

Вначале рассмотрим добавочные соединения, образовавшиеся в процессе упрощения топологии. Вспомним, что эти соединения представляют собой оствное дерево в графе всевозможных соединений, причем за вес ребра в этом графе принимается длина соединения.

По свойству дерева [8], количество его ребер фиксировано: $B - P = 1$, где B — число вершин, P — число рёбер графа. Следовательно, удаление любого из оставшихся соединений приведет к тому, что объект карты перестанет быть односвязным. Значит, условие минимальности количества дополнительных соединений выполняется.

Для выбора минимальных соединений из графа всех возможных соединений использовался алгоритм Прима. Из его корректности [9] следует, что выбранное оствное дерево графа является минимальным. Значит, условие минимальности длин дополнительных соединений выполняется. Таким образом, требование 3 удовлетворено.

Теперь рассмотрим добавочные соединения, образовавшиеся в процессе уменьшения числа вершин.

Пусть дан многоугольник с V вершинами. Тогда описанный выше алгоритм разобьет его на M частей, причем $(M - 1) * N < V + 2 * M \leq M * N$ (слагаемое $2M$ добавляется за счет того, что вершины, через которые проводится разделяющая линия, дублируются в результирующих многоугольниках). Очевидно, что при уменьшении M на единицу количество вершин как минимум одного полученного после разбиения многоугольника будет превышать M ; следовательно, условие минимальности количества дополнительных соединений выполняется.

Одновременная минимизация добавочных соединений и их длины теоретически возможна, но также требует полиномиального времени выполнения. С учетом примечания к требованию 3, решено было ограничиться результатами разработанного алгоритма, который дает оптимальное или близкое к оптимальному решение в большинстве практических случаев.

4.4. Требование 4

Выполнение требования 4 очевидно, т.к. ни один из описанных выше алгоритмов не создавал новых вершин.

4.5. Проверка допустимости соединений

Выше упоминалась проверка проводимых линий на допустимость, т.е. на пересечение с уже существующими линиями. Эта процедура является важной частью алгоритма, т.к. даже одно пропущенное пересечение делает непригодным весь выходной файл.

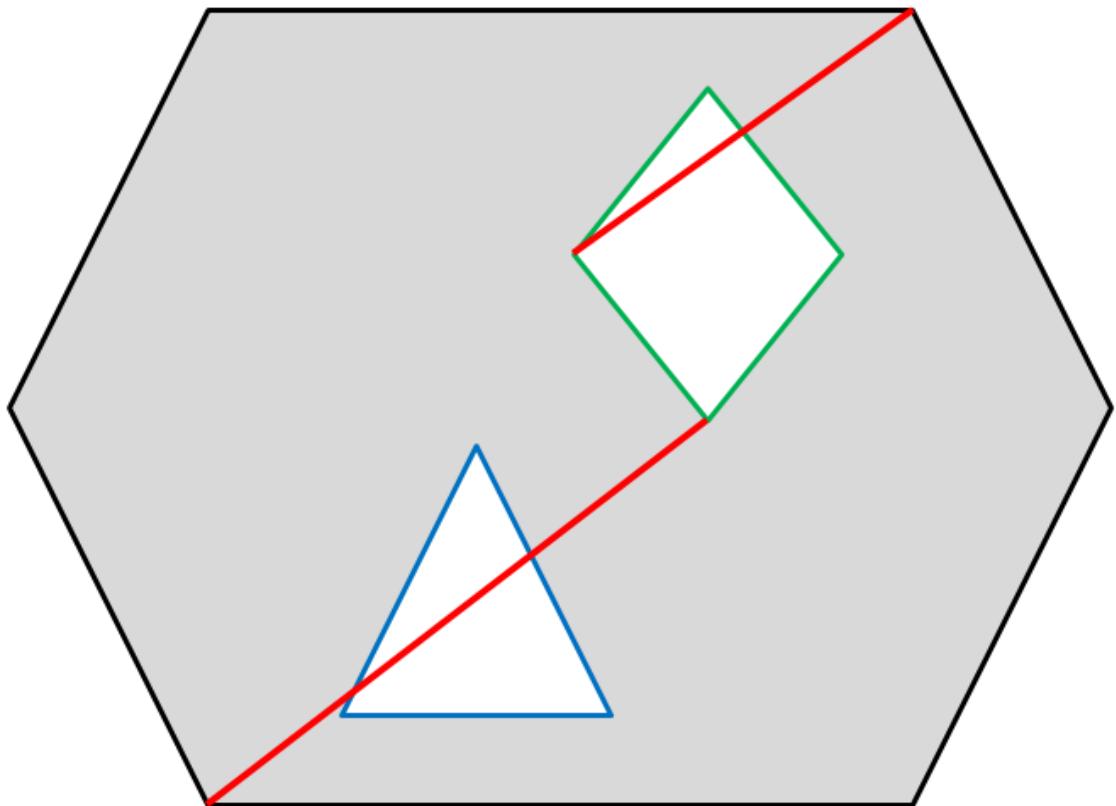


Рис. 9. Примеры недопустимых соединений (выделены красным)

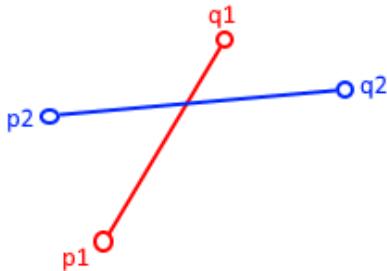
Единственный способ убедиться в отсутствии недопустимых соединений — проверять каждое проводимое соединение на пересечение с каждой из линий, составляющих границы области. Для этого используется алгоритм, описанный в [11].

Этот алгоритм основывается на понятии ориентации. Ориентация упорядоченного списка из трех точек в зависимости от их взаимного расположения может принимать три значения:

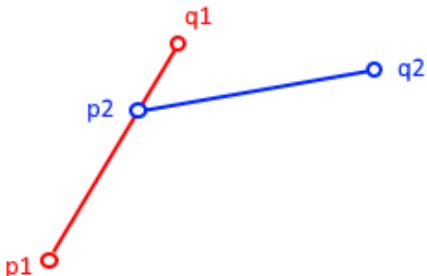
- по часовой
- против часовой
- лежат на одной прямой

Два отрезка, $(p1, q1)$ и $(p2, q2)$, пересекаются тогда и только тогда, когда верно одно из условий (см. рис. 10):

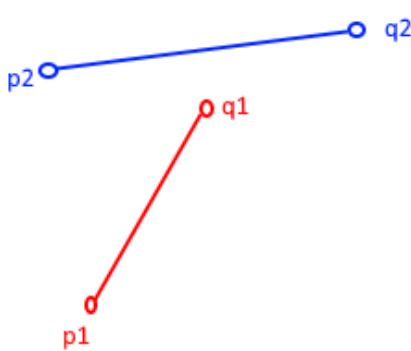
- $(p1, q1, p2)$ и $(p1, q1, q2)$ имеют разную ориентацию
- $(p2, q2, p1)$ и $(p2, q2, q1)$ имеют разную ориентацию



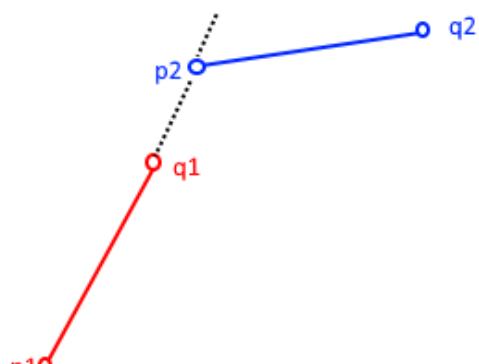
Ориентации $(p1, q1, p2)$ и $(p1, q1, q2)$
различаются. Ориентации $(p2, q2, p1)$
и $(p2, q2, q1)$ различаются.



Ориентации $(p1, q1, p2)$ и $(p1, q1, q2)$
различаются. Ориентации $(p2, q2, p1)$
и $(p2, q2, q1)$ различаются.



Ориентации $(p1, q1, p2)$ и $(p1, q1, q2)$
различаются. Ориентации $(p2, q2, p1)$
и $(p2, q2, q1)$ совпадают.



Ориентации $(p1, q1, p2)$ и $(p1, q1, q2)$
различаются. Ориентации $(p2, q2, p1)$
и $(p2, q2, q1)$ совпадают.

Рис. 10. Возможные ориентации отрезков в общем случае

Отдельно следует рассмотреть случай коллинеарных отрезков. Они пересекаются, если одновременно выполняются условия (см. рис. 11):

- (p_1, q_1, p_2) , (p_1, q_1, q_2) , (p_2, q_2, p_1) , и (p_2, q_2, q_1) лежат на одной прямой
- проекции на ось x отрезков (p_1, q_1) и (p_2, q_2) пересекаются
- проекции на ось y отрезков (p_1, q_1) и (p_2, q_2) пересекаются

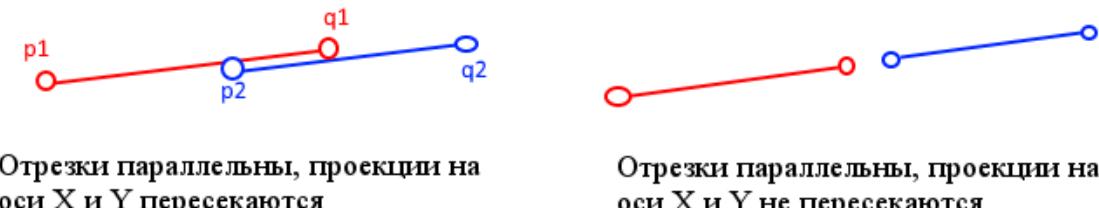


Рис. 11. Возможные ориентации коллинеарных отрезков

Докажем теперь, что проверять сами соединения на пересечения друг с другом не требуется. Для 4.2 утверждение очевидно: описанный алгоритм на каждом шаге исследует лишь одно соединение.

В 4.1 после построения полного графа допустимых соединений возникают пересечения. Однако, после выполнения алгоритма Прима они исчезнут. Почему? Предположим противное: в минимальном оствовном дереве имеются ребра, соответствующие пересекающимся соединениям. Поменяем концы этих соединений таким образом, чтобы не нарушить связность графа (это всегда можно сделать, т.к. количество ребер графа не изменяется). Окажется, что длина каждого из полученных соединений меньше, чем у первоначальных. Но это противоречит корректности алгоритма Прима. Тем самым, утверждение доказано.

5. Реализация алгоритма

Требования, предъявленные к разрабатываемому исполняемому файлу:

1. Время исполнения для предоставленных файлов-примеров имеет порядок секунд или минут. Возможность работы в реальном времени не требуется.
2. Целевая платформа — Microsoft Windows. Доступны инструменты .NET Framework.
3. Графический интерфейс не требуется, т.к. взаимодействие программы с пользователем минимально.

Кроме того, для работы с большими массивами структур данных требуются инструменты высокого уровня. Исходя из этих требований, для реализации алгоритма был выбран язык C#: он полностью поддерживается платформой .NET Framework и обладает высокоуровневым инструментарием LINQ. В качестве среды разработки выбрана Microsoft Visual Studio 2013.

C# относится к семье языков с С-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML [12].

Language Integrated Query (LINQ) — проект Microsoft по добавлению синтаксиса языка запросов, напоминающего SQL, в языки программирования платформы .NET Framework. Используя некоторые новые особенности языка, LINQ позволяет использовать SQL-подобный синтаксис непосредственно в коде программы, написанной, например, на языке C# [13].

5.1. Интерфейс

Выбор файлов для обработки реализован при помощи вызова системного диалога Windows через класс OpenFileDialog .NET Framework. После этого вывод отладочной информации осуществляется через текстовый интерфейс.

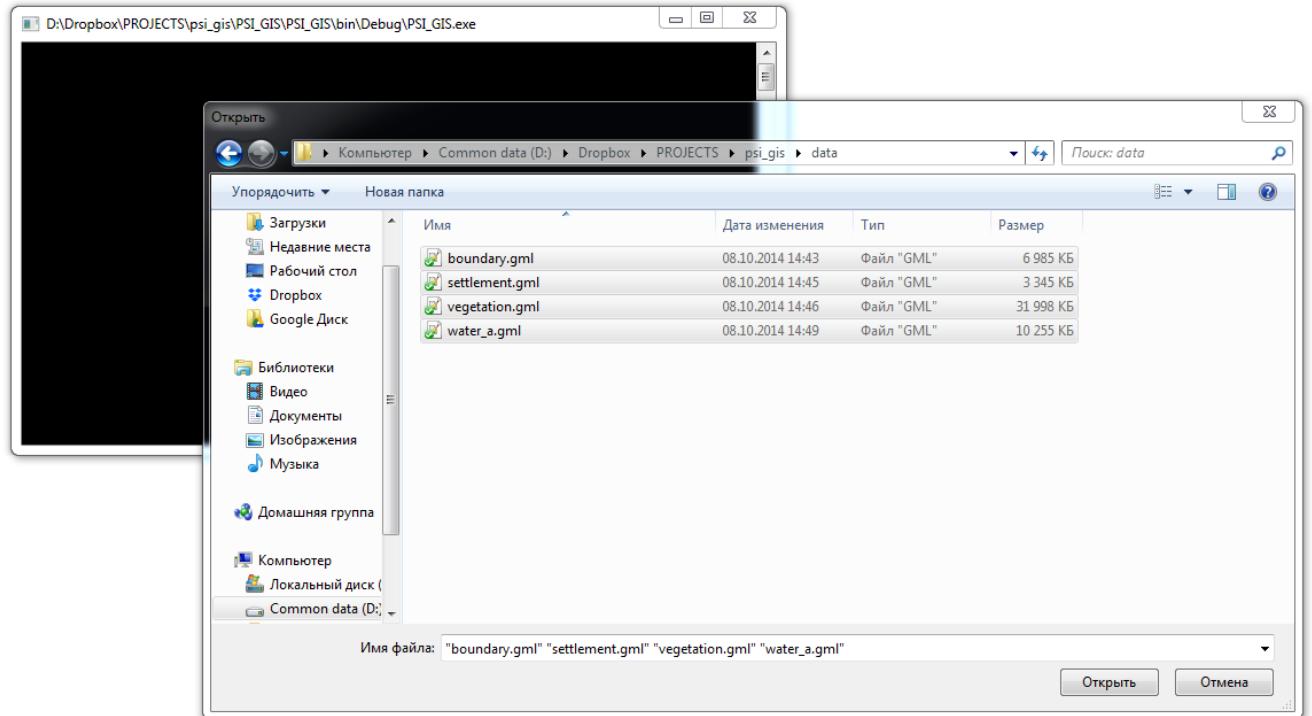
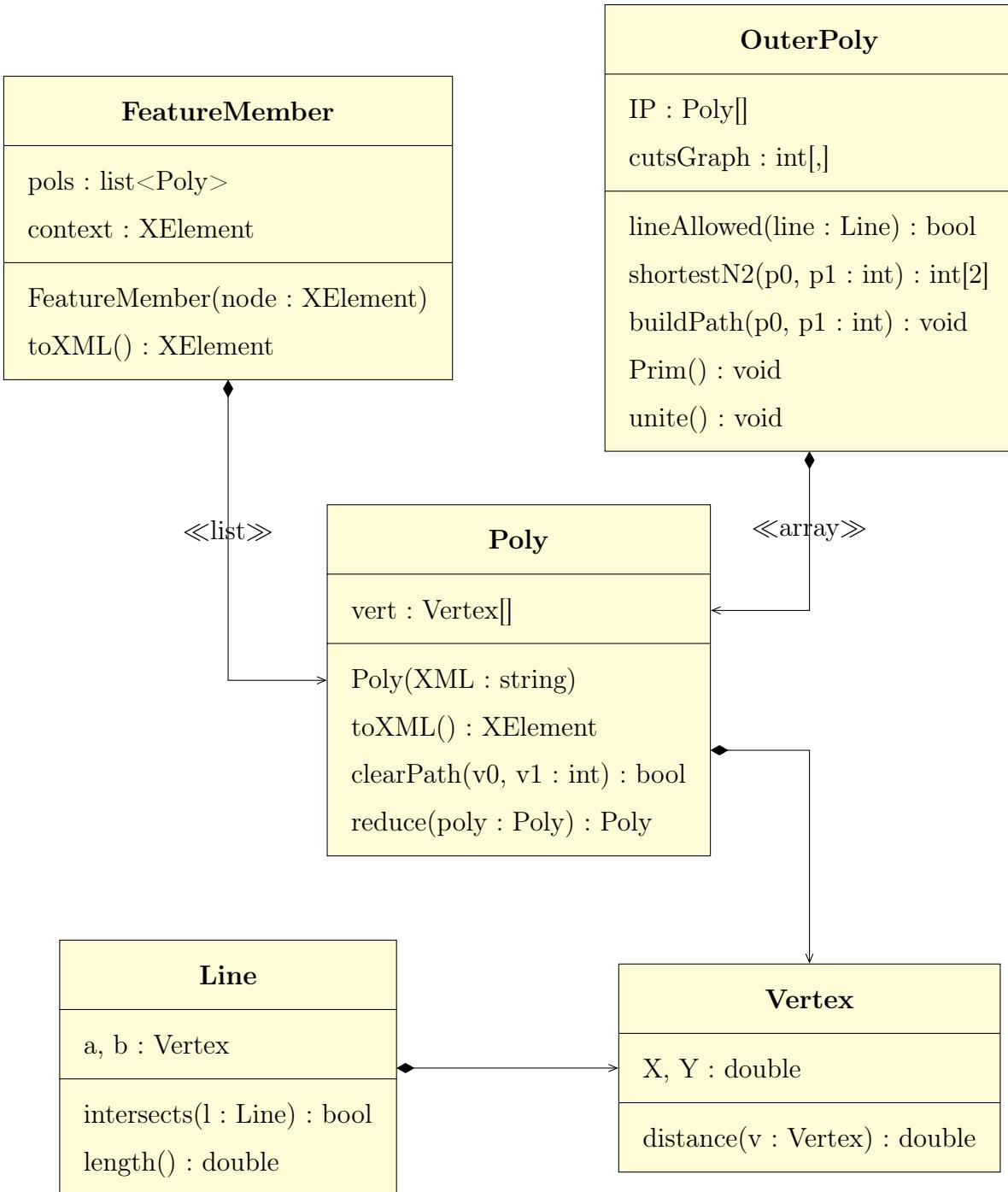


Рис. 12. Интерфейс ввода данных

5.2. Объектная модель



- **Vertex**: базовый класс, представляет точку на плоскости.
- **Line**: представляет отрезок. Состоит из двух объектов **Vertex**. Содержит методы для измерения длины и проверки пересечения с другим отрезком.
- **Poly**: Общее представление многоугольника. Состоит из объектов **Vertex**. Реализует методы прямого и обратного преобразования в GML-элемент

(Poly, toXML). Метод ClearPath проверяет, можно ли соединить две вершины многоугольника, не пересекая при этом его сторон. Метод reduce реализует алгоритм, описанный в [4.2](#).

- **OuterPoly:** Представляет многосвязную область. Состоит из внешней границы, представляемой объектом Poly, и массива многоугольников (IP), представляющих «дыры». Метод lineAllowed проверяет, можно ли провести внутри области отрезок, не пересекающий сторон ни одного из составляющих ее многоугольников. Метод shortestN2 определяет кратчайший отрезок, соединяющий два многоугольника внутри области. Метод unite реализует алгоритм, описанный в [4.1](#) при помощи методов buildPath для построения полного графа соединений и Prim для алгоритма Прима.
- **FeatureMember:** хранит метаданные областей, не имеющие отношения к их геометрии. Реализует ввод и вывод в GML.

5.3. Пример работы

См. рис. [13](#) и [14](#).

Кроме того, в приложении к пояснительной записке присутствуют образцы данных для различных видов географических объектов в формате GML (папка *sample_data*) и исходный код работы (папка *code*), который можно протестировать на этих данных.

6. Заключение

В рамках данной работы была разработана программа для программно-технического комплекса (ПТК) Автоматизированной системы диспетчерского управления (АСДУ) электрическими сетями PSIcontrol по оптимизации геоинформационных полигональных (замкнутых) структур данных на основе математической обработки входных XML-файлов по следующим критериям:



Рис. 13. Визуализация фрагмента входного файла

1. оптимальное перекрытие островных структур на случай их наличия в глобальной структуре без взаимных пересечений с использованием вырожденных туннелей;
2. оптимальное разрезание полигонов, содержащих значительное количество точек, на более мелкие полигоны, количество точек которых не превышает N.

Результаты работы позволили снять ряд существенных ограничений по внедрению ГИС-системы ПТК АСДУ PSIcontrol, что позволило существенно (на два-три порядка) сократить время предварительной обработки геоинформационных подложек для ПТК.

Результаты работы внедрены в практическую деятельность Департамента

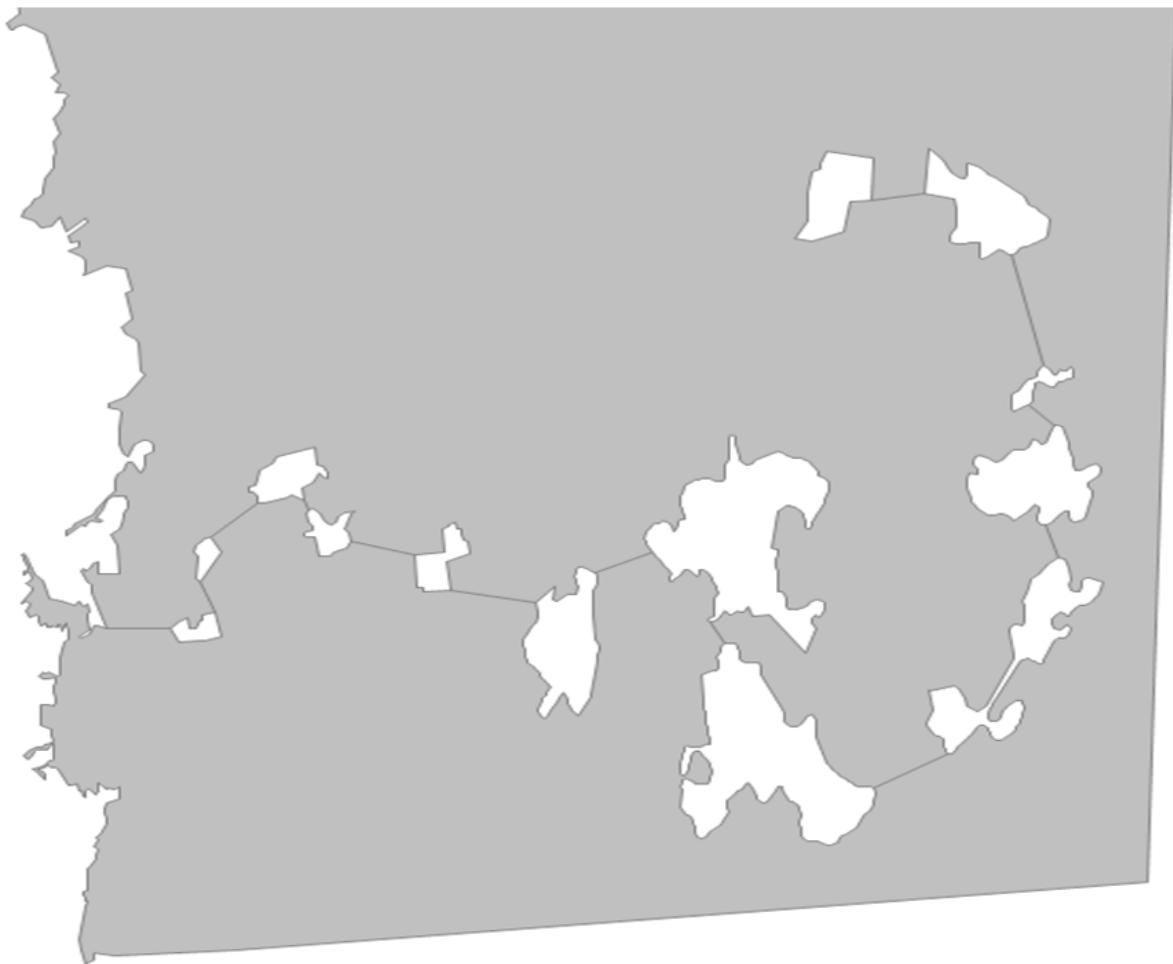


Рис. 14. Результат преобразования фрагмента файла

энергетики ООО «ПСИ» и используются во всех текущих проектах компании по созданию и модернизации АСДУ на предприятиях российской электроэнергетики.

Соответствующий акт о внедрении находится в приложениях.

Список литературы

1. Geography Markup Language | OGC. — URL: <http://www.opengeospatial.org/standards/gml> (дата обр. 19.05.2018).
2. Open Geospatial Consortium (OGC). — URL: <https://www.iotone.com/organization/open-geospatial-consortium/o195> (дата обр. 20.05.2018).
3. ISO 19136:2007 : тех. отч. — URL: <https://www.iso.org/standard/32554.html>.
4. Босс В. ТФКП. Т. 9. — Москва : URSS. — (Лекции по математике). — ISBN 978-5-397-01109-9.
5. *Crowdy D.* A transform method for Laplace's equation in multiply connected circular domains // IMA Journal of Applied Mathematics. — 2015. — Дек. — Т. 80, № 6. — С. 1902—1931. — ISSN 0272-4960, 1464-3634. — DOI: [10.1093/imamat/hxv019](https://doi.org/10.1093/imamat/hxv019). — URL: <https://academic.oup.com/imamat/article-lookup/doi/10.1093/imamat/hxv019> (дата обр. 21.05.2018).
6. Chapter 3: Polygon Triangulation / M. de Berg [и др.] // Computational Geometry (2nd revised ed.) — Springer-Verlag, 2000. — С. 45—61. — ISBN 3-540-65620-0.
7. Элементарная топология / О. Виро [и др.]. — 2-е изд. — МЦНМО, 2012. — ISBN 978-5-94057-894-9.
8. *Олейник Т.* Деревья // Основы дискретной математики: теория и практика. — МИЭТ, 2010.
9. *Prim R. C.* Shortest connection networks And some generalizations // Bell System Technical Journal. — 1957. — 36 (6). — С. 1389—1401. — DOI: [10.1002/j.1538-7305.1957.tb01515.x](https://doi.org/10.1002/j.1538-7305.1957.tb01515.x).

10. *Kruskal J. B.* On the shortest spanning subtree of a graph and the traveling salesman problem // Proceedings of the American Mathematical Society. — 1956. — Янв. — Т. 7, № 1. — С. 48—48. — ISSN 0002-9939. — DOI: [10.1090/S0002-9939-1956-0078686-7](https://doi.org/10.1090/S0002-9939-1956-0078686-7). — URL: <http://www.ams.org/jourcgi/jour-getitem?pii=S0002-9939-1956-0078686-7> (дата обр. 28.05.2018).
11. Introduction to algorithms. — Cambridge, Mass. : MIT Press, 2009. — ISBN 978-81-203-4007-7. — OCLC: 984411568.
12. ISO/IEC 23270:2006 : тех. отч. — URL: <https://www.iso.org/standard/42926.html>.
13. *Cheney J., Lindley S., Wadler P.* A practical theory of language-integrated query //. — ACM Press, 2013. — С. 403. — ISBN 978-1-4503-2326-0. — DOI: [10.1145/2500365.2500586](https://doi.org/10.1145/2500365.2500586). — URL: <http://dl.acm.org/citation.cfm?doid=2500365.2500586> (дата обр. 28.05.2018).