|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Фундаментальные науки

КАФЕДРА Прикладная математика

**ОТЧЕТ ПО ПРАКТИКЕ**

Студент Ануфриев Георгий Алексеевич

*фамилия, имя, отчество*

Группа ФН2-32Б

Тип практики: Ознакомительная практика

Название предприятия: Научно-учебный комплекс «Фундаментальные науки»   
 МГТУ им. Н.Э. Баумана

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ануфриев Г.А.

*подпись, дата фамилия и.о.*

Руководитель практики \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Попов А.Ю.

*подпись, дата фамилия и.о.*

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*2024 г.*

**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технический университет**

**имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

# (МГТУ им. Н.Э. Баумана)

Кафедра «Прикладная математика»

**З А Д А Н И Е**

**на прохождение ознакомительной практики**

на предприятии Научно-учебный комплекс «Фундаментальные науки»   
 МГТУ им. Н.Э. Баумана

Студент Ануфриев Георгий Алексеевич

*фамилия, имя, отчество*

Во время прохождения ознакомительной практики студент должен

1. Изучить на практике основные возможности языка программирования С++ и систем компьютерной алгебры, закрепить знания и умения, полученные в курсах «Введение в информационные технологии», «Информационные технологии профессиональной деятельности».
2. Изучить способы реализации методов решения задачи идентификации положения точки в системе ячеек.
3. Реализовать «метод полос», называемый также «Slab decomposition method».

Дата выдачи задания «26» сентября 2024 г.

Руководитель практики \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Попов А.Ю.

*подпись, дата фамилия и.о.*

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ануфриев Г.А.

*подпись, дата фамилия и.о.*

Оглавление

Постановка задачи

Плоская многоугольная сетка задана следующим образом: дан список точек, каждая из которых задана двумя своими координатами, и список ячеек, для каждой из которых записаны последовательно против часовой стрелки номера вершин из первого списка точек. При этом гарантируется, что любые две ячейки либо не имеют общих точек, либо имеют одну общую вершину, либо примыкают к одному общему отрезку (т.е. имеют общую сторону и две общих вершины).

Требуется идентифицировать положение, т.е. установить принадлежность конкретной ячейке, системы точек, про которые известно лишь то, что они лежат строго внутри ячеек.

а) решить задачу «перебором»;

б) решить задачу эффективно, используя «метод полос», называемый также «Slab decomposition method».

Структура исходного файла данных:

|  |  |
| --- | --- |
| n  x1 y1  …  xn yn  p  m1  k11 k12 … k1(m1)  …  mp  kp1 kp2 … kp(m1)  q  x1 y1  …  xq yq | << количество узлов сетки<< координаты первого узла сетки  << координаты n-го узла сетки<< количество ячеек сетки<< количество углов в первой ячейке<< номера узлов, образующих первую ячейку  << количество углов в p-й ячейке<< номера узлов, образующих p-ю ячейку<< количество точек для идентификации<< координаты первой точки  << координаты q-й точки |

Структура файла результата:

|  |  |
| --- | --- |
| q  c1  …  cq | << количество точек для идентификации<< ячейка, в которую попадает первая точка  << ячейка, в которую попадает q-я точка |

Введение

Основной целью ознакомительной практики 3-го семестра, входящей в учебный план подготовки бакалавров по направлению 01.03.04 – Прикладная математика, является знакомство с особенностями осуществления деятельности в рамках выбранного направления подготовки и получение навыков применения теоретических знаний в практической деятельности.

В ранее пройденном курсе «Введение в специальность» произошло общее знакомство с возможными направлениями деятельности специалистов в области прикладной математики и получен опыт оформления работ (реферата), который полезен при оформлении отчета по практике.

В рамках освоенного курса «Введение в информационные технологии» изучены основные возможности языка программирования С++ и сформированы базовые умения в области программирования на С++. Задачей практики является закрепление соответствующих знаний и умений и овладение навыками разработки программ на языке С++, реализующих заданные алгоритмы. Кроме того, практика предполагает формирование умений работы с системами компьютерной алгебры и уяснение различий в принципах построения алгоритмов решения задач при их реализации на языках программирования высокого уровня (к которым относится язык С++) и на языках функционального программирования (реализуемых системами компьютерной алгебры).

Обзор методов решения задачи

Метод перебора

Самым простым алгоритмом решения задачи является метод перебора. Он заключается в необходимости проверки каждой тестовой точки на принадлежность каждой из возможных ячеек. Принадлежность точки ячейке проверяется методом «учёта числа пересечений» («crossing number algorithm»): из заданной точки в произвольном направлении (например, в положительном направлении горизонтальной оси) выпускается луч, а затем подсчитывается число его пересечений cо сторонами многоугольника. Если оно нечётное, то точка лежит внутри многоугольника, в противном случае – снаружи. Преимущество данного алгоритма состоит в том, что он подходит для проверки точек, принадлежащих как выпуклым, так и невыпуклым многоугольникам.

«Метод полос» («Slab decomposition method»)

Более эффективным алгоритмом решения поставленной задачи является «метод полос». Данный алгоритм, помимо непосредственного определения положения точки в системе ячеек, включает в себя этап предварительной обработки. Он состоит в том, что заданную изначально сетку необходимо разбить на полосы по x-координате следующим образом: каждая граница каждой полосы проходит через вершины ячеек. После этого для каждого многоугольника определяются полосы, в которые он попадает. Делается это c помощью рассмотрения пересечения полосы и диапазона абсцисс текущего многоугольника. Если полученное множество не пустое, то в список, соответствующий данной полосе, добавляется номер ячейки, принадлежащей этой полосе.

На этапе определения номера ячейки, которой принадлежит точка, c помощью бинарного поиска находится полоса, в которой находится данная точка, а затем снова c помощью бинарного поиска среди ячеек, которые попали в данную полосу, определяется та из них, которая содержит данную точку. Для того, чтобы бинарный поиск работал корректно, массивы должны быть отсортированы. Для определения полосы эта задача не представляет трудностей, так как все полосы упорядочены по x-координатам вершин. Для осуществления бинарного поиска по ячейкам в полосе выполняется их сортировка по минимальной ординате ячейки для данной полосы.

В момент определения номера ячейки, которой принадлежит точка, возникает неопределённость в связи c тем, что её ордината может лежать как выше, так и ниже ребра, проходящего через точку c минимальной y-координатой ячейки в этой полосе. Для разрешения этой проблемы составляется уравнение прямой:

где – координаты тестовой точки, – координаты точки на ребре, – координаты точек ребра ячейки c минимальной ординатой в данной полосе, принадлежащих непосредственно полосам. Из уравнения (1) выражается y, после чего проводится сравнение:

* Если , то полученный индекс ячейки в полосе остаётся неизменным;
* Если , то полученный индекс ячейки в полосе уменьшается на единицу.

Описание алгоритмов c использованием псевдокода

Algorithm 1 Метод перебора

Функция ISINTERSECTING(p, a, b)

Входные данные: p – проверяемая точка, из который выпускается горизонтальный отрезок, a, b – точки границ отрезка

Выходные данные: bool T – пересечение выпущенного луча c отрезком [a; b]

1. low = a, high = b – нижняя и верхняя точки отрезка соответственно;

2. if low.y > high.y then меняем местами a и b;

3. if p.y < low.y OR p.y > high.y then return False;

4. if high.y = low.y then return False;

5. Определим потенциальную точку пересечения точки p c отрезком [a; b]:

6. return p.x < x.

Функция RAY\_CAST(p, c)

Входные данные: p – проверяемая точка, c – ячейка, проверяемая на то, что точка p ей принадлежит

Выходные данные: bool T – общее число пересечений выпущенного луча c рёбрами многоугольника нечётно

1. intersections = 0 – общее число пересечений;

2. for (a, b) – пара вершин многоугольника из c

if then ;

end for.

3. return .

Входные данные: P – набор точек, образующих сеть ячеек на плоскости, PNums – набор номеров точек, образующих каждую ячейку, Pts – набор тестовых точек

Выходные данные: CellNums – номера ячеек, которым принадлежат проверяемые точки

0. Инициализируем CellNums;

1. Cells – массив из ячеек;

2. for каждая точка p из Pts

flag = 0 – контроль того, что точка попала в одну из ячеек;

for j := 1…n, где n – размер Cells:

if RAY\_CAST(p, Cells[j]) then:

flag = 1;

append j to CellNums;

break;

end if.

if flag = 0 then add -1 to CellNums – признак того, что точка не попала ни в одну ячейку;

end for.

3. return CellNums.

Algorithm 2 Slab decomposition

Функция YCellInSlab(cell, slab)

Входные данные: cell – ячейка, slab – полоса, в которой лежит ячейка

Выходные данные: Y – набор ординат точек ячейки, принадлежащим границам полосы

1. Инициализируем массивы pair – набор пар вершин, и Y – массив, который будет результатом работы функции;

2. for каждая последовательная пара вершин (a, b) из cell:

if MIN(a.x, b.x) < slab[1] AND (MAX(a.x, b.x) > slab[2] OR MAX(a.x, b.x)=slab[2]) OR MAX(a.x, b.x) > slab[2] AND (MIN(a.x, b.x) < slab[1] OR MIN(a.x, b.x)=slab[1]) OR (MIN(a.x, b.x)=slab[1] AND MAX(a.x, b.x)=slab[2]):

add (a, b) to pair

end if;

end for;

3. for каждая пара p=(a,b) в pair:

for i:=1..2:

add y to Y;

end for;

end for;

4. return Y.

Входные данные: P – набор точек, образующих сеть ячеек на плоскости, PNums – набор номеров точек, образующих каждую ячейку, Pts – набор тестовых точек

Выходные данные: CellNums – номера ячеек, которым принадлежат проверяемые точки

1. Cells – массив ячеек;

2. Инициализируем и заполняем массив setxslab – уникальные x-координаты вершин ячеек;

3. SORT(setxslab);

4. Инициализируем и заполняем массив slabs – набор из пар {setxslab[i], setxslab[i+1]}, где i := 1..N-1, N – размер setxslab;

5. Инициализируем и заполняем массив xRangeTable – диапазон по абсциссе каждой ячейки;

6. Инициализируем и заполняем массив CellsRange так, что CellsRange[i] – массив, содержащий два числа: номера полос, в которых начинается и заканчивается ячейка;

7. i := 1..Slabs.size() – словарь, в котором ключ – номер полосы, значение – список из номеров ячеек, лежащих в этой полосе

8. for i := 1..cellsInSlabs.size():

SortBy(cellsInSlabs[i], MIN(YCellInSlab(cells[a], slabs[i])) &);

end for;

9. Инициализируем CellNums;

10. for каждая точка pt из Pts:

slabInd = BinSearch(setxslab, pt.x);

Создаём массив AllY, который содержит в себе массивы со всеми ординатами каждой ячейки в данной полосе

C помощью массива AllY создаём массив MinY, содержащий минимальные ординаты каждой ячейки в данной полосе

cellInd = BinSearch(MinY, pt.y);

Y = YCellInSlab(Cells[cellInd], slabs[slabInd]);

y1 = MIN(Y[1], Y[3]);

y2 = MIN(Y[2], Y[4]);

if pt.y > y then CellNums[i] = cellsInSlabs[slabInd][cellInd];

else CellNums[i] = cellsInSlabs[slabInd][cellInd - 1];

end for;

11. return CellNums;

Особенности реализации алгоритмов

Реализация метода перебора

Особенности реализации на языке C++

Основным объектом в задаче является точка, поэтому для её хранения была создана структура Point, её атрибутами стали координаты точки. Также была создана структура Cell, которая содержит в себе массив из вершин ячейки. Для динамического хранения большого массива данных было использовано такое средство стандартной библиотеки STL как vector. Также была реализована перегрузка потока вывода для объектов типа std::vector, содержащего в себе любой тип данных, чтобы иметь возможность выводить результат на экран или записывать его в файл.

Исходные данные считываются из файла, во время считывания выполняется проверка на корректность введённых данных. В результате работы программы на экран выводится количество проверяемых точек и номера ячеек, в которые они попали.

Реализация метода перебора

Для реализации метода перебора были реализованы две функции, обе из которых необходимы для реализации метода «учёта числа пересечений». Логическая функция isIntersecting проверяет, пересекается ли горизонтальный луч cо стороной многоугольника. Логическая функция ray\_cast позволяет узнать, принадлежит ли точка многоугольнику. Во время работы в цикле for создаётся переменная flag, изначальное значение которой 0. Она предназначена для того, чтобы отслеживать, попадает ли точка в какую-либо ячейку или нет. Если точка окажется в ячейке, значение переменной flag изменится на 1, и программа выйдет из цикла, отвечающего за проход по всем ячейкам. Если же значение переменной flag останется равным 0, это будет означать, что точка не попала ни в одну ячейку, и в таком случае в вектор результата добавляется -1.

Реализация «метода полос»

Для реализации метода полос c помощью функции xslab заполняется вектор setxslab, предназначенный для хранения x-координат всех точек. Также создаётся вектор из векторов slabs, который заполняется c помощью функции slabsBoundaries и хранит координаты левой и правой границы полосы в каждом своём векторе. В векторе rangeTable хранятся начальные и конечные x-координаты каждой ячейки, которые позже позволяют понять, в какие полосы попал многоугольник и заполнить соответствующий массив cellsRange. Затем заполняется словарь cellsInSlabs, в котором ключи – номера полос, а значения – векторы c номерами ячеек, которые принадлежат данной полосе. После производится сортировка ячеек в полосах по их минимальной y-координате в этой полосе.

После предварительной обработки данных начинается этап поиска положения точки. Для каждой точки вычисляется номер полосы, которой она принадлежит, а затем в данной полосе определяется ячейка, в которой находится эта точка. Все эти действия выполняются c помощью бинарного поиска. Стоит отметить, что в случае, когда искомая точка по y находится выше, чем минимальная y-координата данной ячейки в полосе, но сама точка при этом не принадлежит указанной ячейке, полученный индекс для массива из ячеек в полосе уменьшается на единицу.

Особенности реализации в Wolfram Mathematica

Преимущество системы компьютерной алгебры Wolfram Mathematica для решения поставленной задачи позволяет использовать вспомогательные средства. Так, например, для того чтобы определить, принадлежит ли точка многоугольнику, используется функция RegionMember. Также Wolfram Mathematica обладает широким спектром средств визуализации, что позволяет наглядно представить условие задачи.

Для оптимизации решения задачи списки составлялись параллельными процессами c помощью команды ParallelTable.

Реализация метода перебора

Полный перебор в данной задаче был реализован c помощью ParallelTable, когда каждая точка проверялась на принадлежность каждой ячейке. (что ещё сказать???)

Реализация «метода полос» («Slab decomposition method»)

Для данного алгоритма были написаны две пользовательские функции: search и MyMin. Первая из них реализует функционал бинарного поиска, принимая на вход отсортированный массив, по которому проводится поиск, и одну из координат точки (абсциссу или ординату). Функция MyMin позволяет найти минимальную y-координату ячейки при пересечении c данной полосой, принимая на вход ячейку и полосу как пару граничных значений по x-координате. Для хранения списка номеров ячеек, соответствующих каждой полосе, используется ассоциативный массив cellsInSlabs.