ДИСТАНЦИОННОЕ ОБУЧЕНИЕ

Посов Илья Александрович

РАЗБОР ЗАДАЧ «ЗАДАЧА ОТШЕЛЬНИКОВ» И «МАТЕМАТИЧЕСКОЕ СКАЛОЛАЗАНИЕ» КОНКУРСА КИО-2009

УСЛОВИЕ

Задачи «Задача отшельников» и «Математическое скалолазание» были предложены на конкурсе КИО «Конструируй, Исследуй, Оптимизируй» в 2009 году. Как и во всех задачах конкурса, в них предлагается найти некоторую оптимальную конструкцию, и победителями будут те участники, чье решение окажется наиболее близко к оптимальному. Две рассматриваемые задачи были даны участникам, соответственно, первого и второго уровней, и их условия незначительно различаются. Для начала мы приведем условие задачи «математическое скалолазание».

Предположим, мы расставляем в квадрате несколько точек. При каждой расстановке точек можно найти кратчайшую ломаную, которая соединяет эти точки (пример из шести точек и соединяющей их кратчайшей ломаной изображен на рис. 1). Требуется расставить точки так, чтобы минимальная ломаная была как можно длиннее. Интерфейс программы для решения задачи изображен на рис. 2 и на с. 2 обложки журнала, эта программа позволяет расставлять в квадрате точки, самостоятельно строит минимальную ломаную и затем проверяет, не оказалась ли полученная ломаная длиннее предыдущих. Если да, то полученное решение сохраняется как наилучший результат.

В задаче «Математическое скалолазание» требовалось расставить 16 точек. Оригинальная постановка задачи, поясняющая ее название, звучала так: *однажды две команды устроили соревнование по скалолазанию по необычным правилам. Одна команда забивает в скалу 16 крючьев, а другая соединяет крючья веревкой без пересечений так, чтобы потом как можно скорее пройти по ней весь маршрут. Играя за команду, забивающую крючья, позаботьтесь о том, чтобы даже самый короткий маршрут, проложенный по ним, был как можно длиннее.*

В «Задаче отшельников», вместо 16 точек участники расставляли 19 точек-до-

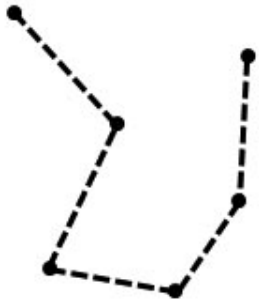

Рис. 1. Минимальная ломаная для шести точек

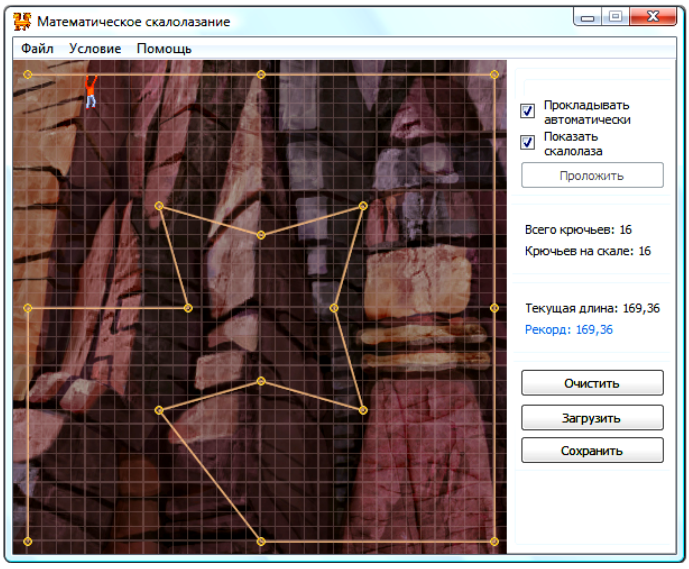

Рис. 2. Интерфейс программы «Математическое скалалазание»

мов, а вместо ломаной, программа строила кратчайшую сеть дорог, соединяющую все дома друг с другом. Интерфейс программы для решения задачи приведенна рис. 3 и на с. 2 обложки. На рис. 4 приведен пример шести точек и минимальной соединяющей их сети дорог. В теории графов такая сеть называется *минимальным остовным деревом*. Оригинальная постановка задачи предлагала расставить дома отшельников так, чтобы максимально затруднить транспортное сообщение между ними, то есть чтобы даже минимальная сеть дорог между домами была как можно длиннее.

Хотя исходная постановка предлагает расставлять в квадрате произвольные точки, в конкурсе задача была сделана дискретной – квадрат был разделен сеткой размера 32×32, и участники могли выбирать только те точки, которые находятся в узлах сетки.

Участникам не требовалось самостоятельно строить минимальные ломаные и минимальные остовные деревья, за них это делала программа. Участники лишь расставляли точки, чтобы получаемые ломаные и деревья были как можно длиннее. Ниже в статье мы обсудим, какие алгоритмы использовались в программе для построения минимальных остовных деревьев и ломаных. Остовное дерево можно запросто построить с помощью простейших классических алгоритмов, а вот построе-

Рис. 3. Интерфейс программы «Задача отшельников»

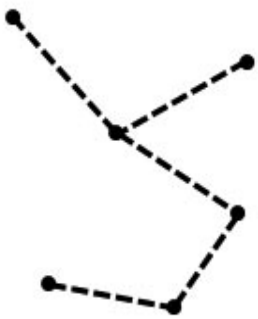

Рис. 4. Минимальное остовное дерево для шести точек

ние минимальной ломаной – нетривиальная задача.

Помимо прочего мы обсудим полученные участниками результаты, посмотрим, как можно было бы искать решение задачи автоматически на компьютере, а для начала попробуем решить задачи в простейших случаях, когда количество точек не превосходит пяти.

РЕШЕНИЕ ЗАДАЧИ С МАЛЕНЬКИМ КОЛИЧЕСТВОМ ТОЧЕК

Можно ли решить предложенные задачи напрямую? То есть привести некоторое расположение точек и доказать, что оно оптимально? Вряд ли, во всяком случае, это совсем не просто. Уже для 5 точек в задаче отшельников решение не очевидно, – мы увидим это ниже. Что же говорить о 16 и 19 точках в задачах из конкурса! Рассуждения усложняет тот факт, что малейшее изменение картины расположения точек может значительно изменить соответственную минимальную ломаную и минимальное остовное дерево. Пример приведен на рис. 5, достаточно сдвинуть одну точку вверх, и минимальная ломаная начинает обходить точки совсем по другому маршруту.

Следовательно, эта задача относится к классу оптимизационных задач, где не приходится надеяться на нахождение точного оптимального решения. Задачу надо решать либо экспериментами с подбором решения (как и было предложено действовать участникам), либо можно запустить автоматические методы оптимизации, изучаемые в разделе информатики с названием Искусственный Интеллект. Они дадут, возможно, не оптимальное решение, но хотя бы близкое к оптимальному. Примерами методов оптимизации могут служить *метод градиентного спуска, метод отжига, генетический алгоритм*. Ниже мы обсудим генетический алгоритм, который несложно реализовать, и он решает задачу, находя при этом решения, похожие на наилучшие решения участников.

Давайте найдем точные решения задачи в случае небольшого числа точек. Если мы расставляем в квадрате две точки, то в обеих задачах оптимальной будет расстановка этих точек в противоположных углах квадрата. Действительно, главная диагональ – самый длинный отрезок внутри квадрата. При расстановке трех точек, видимо, оптимальным решением обеих задач будет поставить две точки в вершины одной из сторон квадрата, а третью точку – на середину противоположной стороны. В случае четырех точек оптимальное решение очевидно – это расстановка точек в вершинах квадрата.

А что произойдет, если точек 5? Тут картина меняется, и задачи отшельника и математического скалолазания имеют разные оптимальные расположения точек. Будем искать решение среди расположений, в которых 4 точки находятся в вершинах квадрата. Остается подобрать позицию для пятой точки. В случае задачи о минимальной ломаной несложно понять, что пятая точка должна находиться ровно в центре квадрата, это изображено на рис. 6.

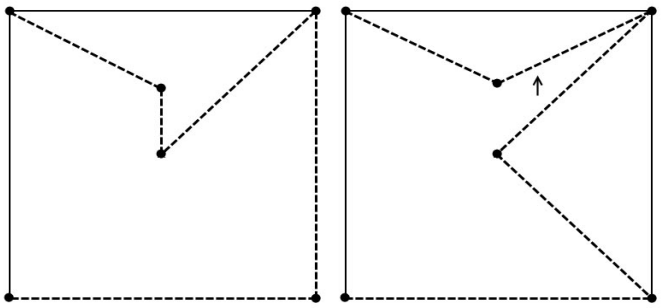

Рис. 5. Сдвиг одной точки вверх значительно изменяет маршрут минимальной ломаной

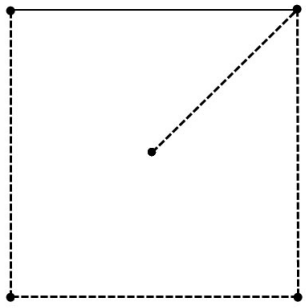

Рис. 6. Оптимальное решение для пяти точек

Почему точка находится в центре? Действительно, минимальная ломаная устроена так, что она проходит по трем сторонам квадрата, а потом соединяет одну из вершин с точкой внутри. Минимальная ломаная будет соединять внутреннюю точку с ближайшей вершиной квадрата, и, располагая точку в центре, мы удаляем ее, насколько возможно, от всех вершин квадрата.

А что если решать задачу отшельников, где вместо ломаной строится минимальное дерево? Поместим точку в центр. Минимальное дерево – это 4 отрезка, соединяющие внутреннюю точку с четырьмя вершинами квадрата. Несложно проверить, что при небольшом сдвиге внутренней точки, устройство минимального остовного дерева не изменяется, то есть оно продолжает состоять из четырех отрезков, идущих из центра в вершины. Но вот длина этого дерева увеличивается.

Итак, расположение с находящейся в центре пятой точкой не оптимально. Но где тогда должна находиться пятая точка? Мы приведем ответ и, вместо доказательства, остановимся на рассуждениях, поясняющих, откуда берется такой неожиданный ответ.

Нам необходимо вспомнить алгоритм построения минимального остовного дерева. Алгоритмы Прима и Краскала хорошо описаны в литературе, и мы не будем подробно повторять их устройство. В двух словах алгоритм Краскала устроен следующим образом: выбираем пару самых близких точек и соединяем их отрезком. Далее выбираем другую пару самых близких точек из еще не соединенных и соединяем их отрезком. Продолжаем процесс, пока все точки не будут соединены в единую сеть. Если в процессе оказывается, что надо провести отрезок, который замыкает цикл из отрезков, то этот отрезок пропускается и не добавляется в сеть.

Как будет работать алгоритм Краскала в нашем случае? Посмотрим на четыре отрезка, ведущие из центральной точки в вершины. Те из отрезков, которые короче стороны квадрата, будут добавлены в дерево в первую очередь. Те из отрезков, которые длиннее стороны квадрата добавлены не будут, а вместо них в дерево попадут сами стороны. На рис. 7 проведены окружности с центром в вершинах квадрата и радиусом, равным длине стороны квадрата. Они разбивают квадрат на области, в которых, исходя из вышесказанного, минимальные остовные деревья устроены одинаково. Действительно, область внутри окружности – это множество точек, расстояние от которых до соответствующей вершины меньше стороны квадрата.

Чтобы найти теперь оптимальную точку, необходимо рассмотреть каждую из областей в отдельности, например, посмотрим на самую маленькую из областей, ту, которая прилегает к стороне квадрата. Оптимальным для этой области будет выбор точки, для которой сумма расстояний до двух соответственных вершин максимальна. Очевидно из рисунка, что оптимальной точкой будет та, которая находится в вершине области (обозначена кружком). Для средней области необходимо выбрать точку, сумма расстояний до всех четырех вершин из которой максимальна. Можно проверить, что это также вершина облас-

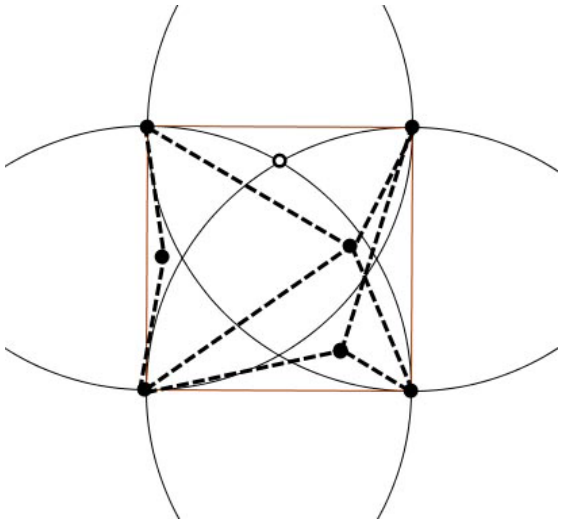

Рис. 7. Области, в которых минимальные остовные деревья устроены одинаково

ти, например точка обозначенная кружком. Для оставшейся области, прилегающей к вершине, необходимо максимизировать сумму расстояний до трех соответствующих вершин. Можно проверить, что опять получается вершина области, то есть точка, обозначенная кружком. Итак, решение задачи – это точка, обозначенная кружком, на рис. 8 она изображена более детально.

На рис. 8 обозначены равные отрезки, обозначен угол в 60°, получающийся как угол равностороннего треугольника. Также обозначен угол в 15° (проверьте, что он действительно равен 15°).

Итак, мы привели ответ для задачи отшельников в случае пяти точек. Он оказался нетривиальным и отличающимся от ответа для задачи о скалолазе в случае пяти точек.

РЕЗУЛЬТАТЫ УЧАСТНИКОВ

Теперь приведем результаты участников, которые они получили, решая задачу отшельников с 19 точками (первый уровень) и задачу скалолаза с 16 точками (второй уровень). Из почти 600 участников первого уровня оптимальное решение получили 2 человека. Это сеть дорог длины ~169,05 (размер квадрата считается равным 32 на 32). Это решение приведено на обложке. Результаты участников распределились равномерно:

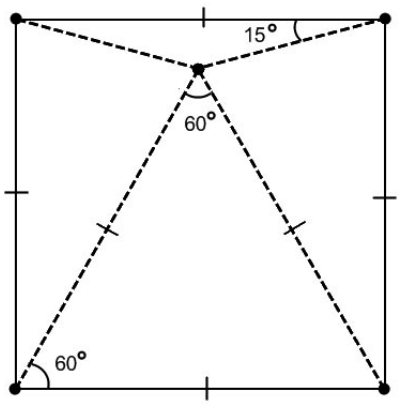

Рис. 8. Решение задачи отшельников для пяти точек

Длина более 169	2 человека
От 168 до 169	11 человек
От 167 до 168	27 человек
От 166 до 167	22 человека
От 165 до 166	34 человека
От 164 до 166	24 человека
От 163 до 164	20 человек
От 162 до 163	25 человек
От 161 до 162	30 человек
161 и менее	400 человек

Является ли полученное участниками решение с длиной 169,05 оптимальным? Возможно. Во всяком случае, жюри не смогло получить решение лучше.

В задаче второго уровня «математическое скалолазание» результаты выглядели следующим образом: из 440 участников лучшее решение получили 44 человека. Это ломаная длины ~169,36. Решение изображено на обложке. Далее приведено распределение результатов оставшихся участников:

Длина более 169	57 человек
От 168 до 169	2 человека
От 167 до 168	8 человек
От 166 до 167	11 человек
От 165 до 166	6 человек
От 164 до 166	23 человека
От 163 до 164	21 человек
От 162 до 163	23 человек
От 161 до 162	18 человек
161 и менее	270 человек

ПОСТРОЕНИЕ МИНИМАЛЬНОЙ ЛОМАНОЙ

Программа решения задачи о математическом скалолазании самостоятельно строила минимальную ломаную, проходящую через заданные точки. Эта задача является NP-трудной, и хорошего алгоритма для нее не существует. Действительно, участники могли заметить, что каждый раз, когда программе необходимо было построить ломаную, она замирала на долю секунды.

Задача напоминает известную NP-трудную задачу коммивояжера. В задаче коммивояжера даны несколько городов, заданы стоимости проезда между каждой

парой городов и требуется построить замкнутый маршрут, имеющий минимальную стоимость и проходящий по всем городам ровно по одному разу. Наша задача отличается тем, что, во-первых, маршрут надо строить не замкнутый, а во-вторых, цены между городами заданы не произвольно, а являются обычными (евклидовыми) расстояниями между соответствующими точками. Все эти изменения не упрощают задачу, и она остается NP-трудной.

Как же все-таки построить минимальную ломаную? Мы рассмотрим три алгоритма: полный перебор возможных ломаных, метод ветвей и границ, а также динамическое программирование. Программа конкурса КИО использовала третий метод, его мы обсудим подробно, а об остальных скажем только несколько слов.

ПОЛНЫЙ ПЕРЕБОР

Самый простой алгоритм, который приходит в голову, это перебрать все возможные ломаные. Сколько их? Каждая ломаная соответствует некоторой перестановке точек, и если у нас есть 16 точек, то необходимо перебрать очень много вариантов, а именно 16! = 20 922 789 888 000. Компьютер способен проделать несколько десятков миллионов простых вычислений за незаметный отрезок времени, но никак не 20 триллионов вычислений. Можно заметить, что на самом деле ломаных в два раза меньше, потому что ломаная не имеет направления, и две перестановки, отличающиеся чтением с конца в начало, задают одну и ту же ломаную. Но это наблюдение не помогает сделать перебор осуществимым.

МЕТОД ВЕТВЕЙ И ГРАНИЦ

Другой подход, традиционно используемый для задач, близких задаче коммивояжера, – это метод ветвей и границ. Программа конкурса КИО не использовала метод ветвей и границ только потому, что время его работы непредсказуемо и может значительно изменяться. В удачной ситуации программа будет считать долю секунды, в неудачной – до нескольких секунд, что неприемлемо. Несмотря на эту особенность, в реальных задах используется именно метод ветвей и границ, потому что он оказывается осуществим для большого числа точек, в отличие от динамического программирования, которое осуществимо только если точек не более 16 или 17.

Попробуем описать в общих чертах работу метода ветвей и границ. Чтение этой части потребует от читателя более глубокого знакомства с теорией графов и алгоритмов. Итак, пусть надо провести ломаную через несколько точек. Рассмотрим граф, вершины которого – это точки, и каждая пара вершин соединена ребром, вес которого равен расстоянию между точками. Заметим, что путь через все вершины является остовным деревом, поэтому, если мы построим минимальное остовное дерево графа, мы получим оценку снизу длины пути. (Другими словами, любой путь через все вершины не короче минимального остовного дерева).

Итак, построим минимальное остовное дерево. Если оно образует путь – задача решена. Если нет, то найдется вершина, из которой выходит минимум 3 ребра – e_1, e_2, e_3, ... Мы знаем, что в искомом пути не могут остаться все эти ребра, поэтому представим все множество возможных ломаных в виде объединения нескольких подмножеств. В первое подмножество войдут ломаные, которые не содержат ребро e_1, во второе подмножество – те ломаные, которые не содержат ребро e_2, и так далее. Разделение некоторого множества решений на несколько подмножеств называется ветвлением.

Затем мы можем оценить длину минимальной ломаной в каждом из подмножеств, для чего опять надо построить минимальные остовные деревья, учитывая, конечно, что некоторые ребра отсутствуют, и их проводить нельзя. Пусть, для примера, в первом подмножестве получилось дерево размера 10, во втором – 20, а в третьем – 30. При этом пусть дерево во

втором подмножестве является путем. Получается, что существует ломаная длины 20. Третье множество решений нас больше не интересует, потому что в нем никак не получится найти ломаную короче 30. А вот в первом множестве, возможно, есть решение. Это множество решений надо ветвить дальше, то есть снова разделить на несколько подмножеств.

Реализации метода ветвей и границ различаются тем, в каком порядке выбираются множества для ветвления. В приведенном примере выбора не было, оставалось только одно множество для продолжения работы. Но вот если бы нам не повезло, и дерево из второго множества не оказалось бы путем, пришлось бы выбирать, какое из трех полученных множеств ветвить. Очевидная эвристика – ветвить то множество, которое предлагает самую маленькую оценку на размер решения. Это возможная стратегия, но она требует хранить в памяти все построенные на данный момент множества. Другие стратегии действуют методами, близкими к поиску в глубину, поэтому намного менее требовательны к памяти, но и ответ при этом находят не так быстро.

ДИНАМИЧЕСКОЕ ПРОГРАММИРОВАНИЕ

Для построения кратчайшей ломаной было использовано динамическое программирование. Обозначим множество всех точек за M, обозначим за $D(U, u)$ – длину кратчайшей ломаной, проходящей через все точки множества U и начинающейся в точке u. Здесь U – это несколько точек, то есть подмножество M, а u – одна из точек подмножества U.

Как вычислить D для всех возможных значений параметров? Во-первых, очевидным образом вычисляются $D(U, u)$ для U, состоящих из одного элемента. Для таких значений $D(\{u\}, u) = 0$. Далее, чтобы вычислить $D(U + v, v)$, где плюс обозначает добавление точки к множеству, необходимо попробовать соединить точку v последовательно со всеми точками множества U и выбрать минимальную из получающихся ломаных. Формально:

$D(U + v, v) = \min_{u \text{ из } U} \{D(U, u) + \text{dist}(u, v)\}$.

Следовательно, последовательно увеличивая размер множества U, можно вычислить значения D для всех возможных параметров.

Остается определить кратчайшую ломаную через точки множества M. Выберем из $D(M, m_1)$, $D(M, m_2)$, $D(M, m_3)$ ... самое маленькое число (здесь m_i – это точки множества M). Это число и есть длина самой короткой ломаной. Далее, ломаную можно восстановить. Начинается она, естественно, в том самом m_i, для которого $D(M, m_i)$ оказалось минимальным. Следующая вершина восстанавливается перебором, т.е. надо попробовать соединить m_i со всеми оставшимися точками и посмотреть, в каком из случаев длина соответствующей ломаной окажется совпадающей с длиной минимальной ломаной. После того как найдены две точки ломаной, аналогично находится третья точка, и процесс продолжается, пока не будут найдены все точки ломаной.

Оценим количество памяти, которое требуется для построения минимальной ломаной из n точек. Количество подмножеств U множества M равно 2^n. Но у $D(|U|, u)$ есть второй параметр, который принимает в худшем случае одно из n значений. Следовательно, для работы программы необходимо хранить порядка $n2^n$ вещественных чисел. Если учесть, что вещественные числа занимают в памяти примерно 10 байтов, то получается, для работы программы с 16 точками необходима память около 10 мегабайт. В случае 17 точек – 20 мегабайт, а 18 точек – почти 50 мегабайт.

МЕТОДЫ ОПТИМИЗАЦИИ

Как уже было написано, вряд ли задачи задачи отшельников и скалолазов можно решить точно. В подобных случаях, вместо оптимальных решений, интересно получить хотя бы решения, близкие к оптимальным. Именно это сделали участники конкурса, но как можно было бы получить решения автоматически?

Генетический алгоритм – это один из многочисленных методов оптимизации. В данной задаче он интересен тем, что его нетрудно реализовать, при этом он сразу же выдает результаты, хоть и не превосходящие лучшие решения участников, но очень близкие к ним.

Метод основан на механизмах, напоминающих биологическую эволюцию. Нам необходимо подобрать набор из точек с самой длинной минимальной ломаной, для этого создадим некоторое количество случайных наборов точек и назовем это первоначальной популяцией. Далее наборы будут скрещиваться, мутировать и удаляться из популяции в соответствии с принципами естественного отбора. В конце останутся наборы, которые дают самую длинную минимальную ломаную.

После первого шага создания случайной первоначальной популяции на втором шаге наборы точек надо скрестить. Скрещивание – это процесс, при котором два набора – родители дают еще один набор – ребенка. Необходимо определиться с тем, что такое скрещивание в нашем случае. Будем считать, что ребенок – это набор, в котором около половины точек взяты от первого родителя, а остальные точки – от второго. В скрещивании участвуют не все наборы, а только лучшие, то есть те, которые соответствуют самой длинной минимальной ломаной. Допустим, участвуют только верхние две трети популяции, а нижняя треть не выживает.

Новая популяция содержит только те точки внутри наборов, которые были раньше. Чтобы не ограничиваться небольшим количеством точек, каждый раз будем производить мутацию, при которой случайным образом отбираются некоторые наборы и в них случайные точки передвигаются.

После этого процесс надо снова повторить и снова скрестить наборы друг с другом. Аналогия происходящего с эволюционными процессами очевидна. Какой объект выступает в данном случае в роли генов? Гены набора – это его точки. В процессе отбора остаются те точки-гены, которые соответствуют более полезным признакам наборов. Например, те точки, которые находятся в углах квадрата, полезны для любого набора, с ними ломаная становится длиннее. В процессе отбора в какой-то момент оказывается, что большинство наборов содержат все четыре угловые точки. Заметим, что мы не вкладывали этот факт в алгоритм, он вывелся сам собой в процессе отбора.

К примеру, данным методом в задаче первого уровня был получен ответ с длиной ~168,20, выглядит он так же, как и лучший ответ участников с длиной 169,05, но некоторые точки сдвинуты чуть в сторону, и поэтому ответ не такой большой, каким мог бы быть.

Посов Илья Александрович,
ассистент кафедры ВМ–2
СПбГЭТУ «ЛЭТИ».