АЛГОРИТМ РЕШЕНИЯ ЗАДАЧИ КОММИВОЯЖЁРА МЕТОДОМ ДВИЖЕНИЯ ОТ ПЕРИФЕРИИ

Шуть В. Н., Дронь А. Н. (Брестский филиал Института современных знаний им.А.М. Широкова), Бирук А. С. (Брестский государственный технический университет)

Знаменитая задача коммивояжера была поставлена еще в 1834 году. В своей области (оптимизация дискретных задач) задача коммивояжера служит своеобразным полигоном, на котором испытываются все новые методы.

Классическая постановка задачи о коммивояжере выглядит следующим образом. Имеется N городов, которые должен обойти коммивояжер с минимальными затратами. При этом на его маршрут накладывается два ограничения: маршрут должен быть замкнутым, то есть коммивояжер должен вернуться в тот город, из которого он начал движение; в каждом из городов коммивояжер должен побывать точно один раз.

Метод движения от периферии к центру.

Т.к. оптимальный алгоритм решения задачи коммивояжера еще не найден, предпримем попытку разработать метод, приводящий к более оптимальному результату чем уже известные методы.

Алгоритм данного метода содержит следующие этапы:

1) поиск и последовательное включение в маршрут городов, находящихся в самых отдаленных точках местности.

Первоначально, определяется самый западный город, и включается в маршрут. Далее поочередно находятся крайние северный, восточный и южный города.

Включение городов в маршрут происходит путем определения направленности дуги. Дугой будем называть часть пути, соединяющего два города. Кроме того, для дуги рассчитывается ее длина по следующей формуле: $l = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ где x1, y1, x2, y2 – координаты точек F1(x1,y1) и F2(x2,y2), l – расстояние между точками (рис.2а).

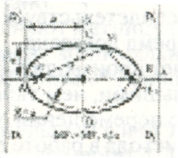

Рис. 1. Эллипс и его параметры

Здесь и далее будем учитывать тот факт, что при рассмотрении задачи коммивояжера на плоскости решение ее будет представлять собой орграф, города – точки и т.д. Т.е. будут употребляться математические эквиваленты некоторых понятий.

В результате имеем замкнутый контур с точками внутри его и снаружи.

2) поиск последовательно для каждого ребра городов находящихся, вне многоугольника текущего маршрута и при этом как можно дальше от

ребра. Дальность рассчитывается путем сложения расстояний от точки до границ ребра. Затем самая дальняя точка разрывает старое ребро, и на его место вставляются два новых. Таким образом, поступаем со всеми старыми ребрами. Данная операция проводится 2 раза.

3) сортировка дуг по их длинам по возрастанию. Это связано с тем, что поиск новых городов будет проводиться, начиная с самой маленькой дуги, что должно дать наименьшее искажение маршрута.

4) поиск и включение города ближайшего к наименьшему ребру.Для наименьшего ребра осуществляется поиск города, который расположен ближе всего к данному ребру и при этом попадает в область эллипса с граничными точками ребра, представляющими собой фокусы эллипса. Попадание города в эллипс ребра определяется следующим неравенством:

$\frac{l_1 + l_2}{3} \le \frac{l}{2}$ (при эксцентриситете равном $\frac{2}{3}$)где l_1 и l_2 – расстояния от новой точки до фокусов эллипса, l – расстояние между фокусами эллипса (смотри рис.1).

Если такой город находится, то он разрывает старое ребро, и на его место добавляются два новых ребра. Затем следует переход на этап 3 с новым множеством ребер.

Может возникнуть ситуация, когда в область площади эллипса не попадет ни одна точка. Тогда переходим к рассмотрению самого малого из оставшихся ребер и выполняем еще раз четвертый этап алгоритма.

Когда не остается точек попадающих в область площади эллипса хотя бы одного ребра, но при этом точки, не попавшие в маршрут, все же остались, происходит переход к этапу 1 для всех городов, не вошедших в маршрут.

Переход к пятому этапу происходит, когда больше не существует свободных городов, и одновременно с этим образовалось более одного замкнутого контура либо нескольких контуров и точка. В результате образуется ряд обособленных контуров (смотри рис.3а).

5) устранение петель в контуре

Просматриваются попарно все дуги одного конкретно контура, и определяется, есть ли между ними пересечение. Для этого в уравнение каждой (y = A * x + B) дуги подставляются поочередно обе крайние точки второй дуги. Если одна из точек имеет значение $y > 0$, а вторая - $y < 0$, то проводится аналогичная проверка, поменяв дуги местами, т.е. точки одной дуги подставляются в уравнение прямой второй дуги.

При обнаружении пересечения оно исправляется следующим образом: из диапазона дуг и точек, относящихся к ним, удаляются пересекающиеся ребра. Далее в контур (на место удаленных) добавляются соответственно ребро, соединяющее первую и предпоследнюю точки «петли» и ребро, соединяющее вторую и последнюю точки «петли».

6) устранение пересечений контуров

Определение пересечений определяется аналогичным пункту 5 образом. Устраняется пересечение путем включения во внутренний контур всех точек, попавших в его область из внешнего контура.

7) объединение контуров

На данном этапе попарно сравниваются пары точек (т.е. получается движущееся «окно» из четырех точек). Сравнение заключается в поиске такой пары дуг, которая давала бы соединение через них двух отдельных маршрутов в один при минимальной сумме длин этих двух дуг.

Оценка метода. Основная трудность, возникающая при решении задач коммивояжера данным методом, связана с необходимостью проведения большого количества математических вычислений, что повышает требования к вычислительной мощности процессора ЭВМ. Вычисления чаще всего связаны с нахождением расстояний между двумя точками, которые проводятся в большом количестве при каждой итерации алгоритма. Определение расстояния требует двух операций умножения и одно вычисление квадратного корня за раз: $l = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

где x_1, y_1, x_2, y_2 – координаты точек $F_1(x_1,y_1)$ и $F_2(x_2,y_2)$, l – расстояние между точками. Кроме того, вычисление условия попадания точки в область эллипса, требует двух операций деления за одну проверку: $\frac{l_1 + l_2}{3} \le \frac{l}{2}$ (при эксцентриситете равном 2/3). ГДЕ l_1 и l_2 – расстояния от новой точки до фокусов эллипса, l – расстояние между фокусами эллипса.

Что касается временных затрат на вычисление то получим следующую зависимость: $t \to O(\frac{1+n}{2} * n)$, при количестве городов равном n. В приведенной формуле учтено время на определение принадлежности точки к области эллипса, и не учтено время на вычисление крайних точек всей области их расположения, образующих выпуклый многоугольник, в связи с незначительностью затрат на их определение. Формула временной зависимости объясняется наличием в алгоритме двух основных циклов. Первый цикл выполняется до тех пор, пока все города не будут включены в маршрут. Второй цикл (вложенный) выполняется для каждого отдельного ребра, т.е. участка пути непосредственно соединяющего только 2 города, и в нем, путем перебора всех не включенных в маршрут городов, определяется город ближайший к текущему ребру и включается в маршрут.

Кроме сказанного выше можно еще отметить, что вычислительный алгоритм достаточно простой и легко программируется на ЭВМ, что является безусловным плюсом при возможной в дальнейшем модернизации алгоритма.

Решение примеров на ЭВМ типа IBM PC показало практическую реализуемость алгоритма и его достаточную эффективность.