РЕШЕНИЕ ЗАДАЧИ НАХОЖДЕНИЯ ОПТИМАЛЬНОГО МАРШРУТА СЛУЖБЫ ПОДДЕРЖКИ СРЕДСТВАМИ ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ

Боднар А.А.
Республиканское высшее учебное заведение
«Крымский гуманитарный университет»,
г. Ялта

В работе предлагается использование метода динамического программирования для нахождения оптимального маршрута службы поддержки с учетом приоритета заказов выполнения провайдеров информационно-коммуникационных сервисов. Выполнена реализация предложенного алгоритма и проведены его испытания на реальных данных.

В фирме провайдера информационно-коммуникационных сервисов есть N заказов для выполнения, расположенных на одной прямой. Сотрудник службы поддержки пронумеровал их числами от 1 до N в порядке увеличения важности. Сотрудник службы поддержки находится в точке 1 и хочет попасть в точку N. Сотрудник службы поддержки может посещать заказ только в порядке увеличения номеров важности. Для перемещения между заказами сотрудник службы поддержки может воспользоваться услугами транспортной компании. Стоимость передвижения от заказа i к заказу j равна $c_i \cdot |x_i - x_j| + t_j$, где x_i – координата заказа i, x_j – координата заказа j, c_i – стоимость единицы передвижения к заказу i, а t_j – стоимость перемещения к заказу j. Необходимо потратить минимум средств на передвижения между заказами.

Будем решать задачу при помощи динамического программирования. Обозначим через $f(i)$ наименьшую стоимость поездки от заказа 1 к заказу i. Ясно, что $f(1) = 1$. Для остальных заказов f вычисляется по следующей формуле:

$$f(i) = \min_{1 \le j \le i} \{ |x_i - x_j| \cdot c_j + t_i + f(j) \}.$$

Основная сложность заключается в том, чтобы вычислить минимум по j быстрее, чем за $O(N)$. Для этого дадим некую геометрическую интерпретацию этой формуле. Вспомним, что любую не вертикальную прямую можно задать уравнением вида $y = k \cdot x + l$, где k и l – это произвольные параметры, x – независимая переменная, а y – зависимая переменная [1].

Пересекать верхнее огибающее множество с вертикальной прямой будем точно также, как и в случае с массивом, только теперь вместо бинарного поиска у нас будет спуск по дереву. Итоговая сложность решения $O(N\ logN)$.

Предложенный алгоритм реализован в программном комплексе единой системы управления службой поддержки провайдера информационно-коммуникационных сервисов.

Литература:

1. Official Solutions ACM ICPC. – Central European Olympiad in Informatics. – Tîrgu Mureş, România. - July 8 – 14, 2009. – 3 p.