УДК 519.178 + 519.977.5
ББК 22.18

# НЕКОТОРЫЕ АЛГОРИТМЫ ОПТИМИЗАЦИИ И ВИЗУАЛЬНОГО ПРЕДСТАВЛЕНИЯ ТРАНСПОРТНЫХ ПОТОКОВ

**Гордийчук Т. В.**[1] **Ицков А. Г.**[2]
(*Ижевский Государственный Технический Университет, Ижевск*)

*Приведен алгоритм поиска оптимального пути на графе с переменными весами. Рассмотрены формулы построения двумерных кубических сплайнов дефекта 1 и их вывод.*

Ключевые слова: взвешенный граф, кратчайший маршрут, кубический сплайн.

## *Введение*

В транспортных сетях крупных городов существует множество маршрутов, и редко удается определить время прибытия, зная время отправления, и наоборот. Возникает проблема реализации алгоритма нахождения кратчайшего пути при условии, что известно расписание движения общественного транспорта.

В дискретной математике существует ряд алгоритмов оптимизации на графе, в частности, алгоритмов нахождения кратчайших расстояний, но в классической теории они реализованы на статических весах, т. е. вес дуги с течением времени не меняется. На практике же, если граф – это модель движения общественного транспорта, время прохождения пути может быть разным. Это обусловлено тем, что вес дуги будет зависеть не только от скорости движения, но и от времени ожидания и стоянки маршрута.

[1] *Годийчук Тарас Владимирович, студент Ижевского Государственного Технического Университета* (*tsimplex@gmail.com*).

[2] *Ицков Александр Григорьевич, кандидат физико-математических наук, доцент* (*pmi.istu@gmail.com*).

Проблема поиска кратчайшего пути не является новой. Существует ряд алгоритмов для решения такого рода задач [2]. В сетевом планировании рассматриваются различные проблемы оптимизации на графах со случайными весами. В исследовании операций приводятся алгоритмы поиска максимального динамического потока [3]. В данной работе рассматривается задача поиска оптимального пути на графах, у которых пропускная способность дуг меняется динамически и зависит не только от времени, но и от управления, под действием которого происходит движение.

## *1. Постановка задачи*

Пусть заданы ориентированный граф $\mathcal{G} = (X, A)$, где $X = \{x_1, x_2, \ldots, x_n\}$ – множество вершин, а $A = \{a_1, a_2, \ldots, a_m\}$ – множество дуг; множество допустимых управлений $U = \{u_1, u_2, \ldots, u_k\}$, и выполнены следующие условия:

1) $\mathcal{G}$ – связный граф и для всех $x_i,\ x_j \in X$ существует путь из $x_i$ в $x_j$;

2) для всех $a_i \in A$ существуют функции $f_i(t, u)$ (время выхода из дуги $a_i$) такие, что $f_i$: $I \times U_i \to I$, где $I = [0, +\infty)$, $U_i \subset U = \{u_1, u_2, \ldots, u_k\}$, где $U_i$ – множество допустимых управлений на дуге $a_i$;

3) для всех $t_0,\ t_1 \in I$ таких, что $t_1 > t_0$ и для всех $u \in U_i$ выполнено $f_i(t_0, u) < f_i(t_1, u)$, $i = 1, 2, \ldots, m$ (при одном и том же управлении – чем раньше произошло попадание в дугу, тем меньше будет время выхода из нее).

Тогда, если заданы начальная $x_{\text{н}}$ и конечная $x_{\text{к}}$ вершины и время начала движения $T_{\text{н}}$ (время окончания движения $T_{\text{к}}$), необходимо найти путь из $x_{\text{н}}$ в $x_{\text{к}}$ – $\{a_j^0\}_{j=1}^l$, значения $\{u_j^0\}_{j=1}^l$ и время $T_{\text{к}}$ ($T_{\text{н}}$), так чтобы $T_{\text{к}} - T_{\text{н}} \to \min$.

При найденных значениях $\{a_j^0\}_{j=1}^l$, $\{u_j^0\}_{j=1}^l$ и известном времени начала движения $T_{\text{н}}$ время окончания движения $T_{\text{к}}$ рассчи-

тывается по следующему алгоритму:

$$
\begin{aligned}
&t_0 := T_{\text{н}},\\
&t_1 := f_1^0(t_0, u_1^0),\\
&t_2 := f_2^0(t_1, u_2^0),\\
&\ldots\ldots\ldots\ldots\ldots\\
&T_{\text{к}} := t_l := f_l^0(t_{l-1}, u_l^0),
\end{aligned}
$$

где $f_j^0$ – вес дуги $a_j^0$.

Рассмотрим поставленную задачу на примере движения общественного транспорта. Пусть известен граф и расписание движения маршрутов. Тогда мы будем рассматривать $U$, как множество всевозможных маршрутов в транспортной сети, а $U_i$ – множество маршрутов проходящих через дугу с номером $i$. Функции $f_i(t, u)$ будут представлять собой сумму времени ожидания маршрута $u$ (которое будет зависеть от времени попадания в начало дуги $a_i$) и времени прохождения дуги на маршруте $u$.

## 2. Алгоритмы

### 2.1. ПОИСК ОПТИМАЛЬНОГО ВРЕМЕНИ ПРОХОЖДЕНИЯ ДУГИ

Пусть $\mathcal{G}$ — исходный граф, $S$ — множество используемых маршрутов, $t_0$ — начальное время, $a_l$ — дуга графа $\mathcal{G}$ (она задается парой чисел $(i, j)$, где $i$ — номер вершины, соответствующей началу дуги, а $j$ — номер вершины, соответствующей концу дуги). $t_{in}^{(kl)}$ — множество времен прибытия $k$-го маршрута в $l$-ую дугу, $t_{out}^{(kl)}$ — множество времен отправления $k$-го маршрута из дуги с номером $l$. Наборы $t_{in}^{(kl)}$ и $t_{out}^{(kl)}$ упорядочены по возрастанию. Найдем в наборе $t_{in}^{(kl)}$ ближайшее к начальному $t_0$ время $t^{(kl)}$, удовлетворяющее условиям:

1) $t^{(kl)} \leqslant t_{in}^{(kl)}[p]$ ($t_{in}^{(kl)}[p]$ — $p$-ый элемент набора);

2) $t_{in}^{(kl)}[p-1] < t^{(kl)}$.

Если не существует такого $t_{in}^{(kl)}[p]$, которое удовлетворяет указанным выше условиям, то $k$-ый маршрут не проходит через эту дугу при начальном времени $t_0$. Зная индекс $p$ находим время $\tau^{(kl)} = t_{out}^{(kl)}[p]$ (оптимальное время прибытия в конец дуги $a_l$ при использовании маршрута с номером $k$).

Для дуги $a_l$ находим все маршруты, которые проходят через нее и попадают в множество используемых маршрутов $S$. Находя для каждого такого маршрута оптимальное время $\tau^{(kl)}$ и минимизируя его по $k$, находим минимальное время выхода $\tau^{(l)}$ из этой дуги и номер маршрута $u_l$, на котором этот минимум достигается.

### *2.2. ПОИСК ОПТИМАЛЬНОГО ПУТИ*

Пусть существует граф $\mathcal{G}$ (граф со взвешенными вершинами), который задан двумя таблицами: таблицей дуг и таблицей вершин, — таблица маршрутов и расписание движения, которое связывает все эти таблицы. Заданы начальное время $t_0$ и две вершины $x_{in}$, $x_{out}$ (где $x_{in}$ — вершина отправления, $x_{out}$ — вершина прибытия, а $t_0$ — время отправления). Построим неявно по этим данным граф $\mathcal{G}'$ — граф с тем же множеством вершин, но с постоянными весами дуг. Обозначим через $C_l^{(k)}(t_{in}, t_{out})$, а через $C'_l$ веса дуг $a_l$ и $a'_l$ графов $\mathcal{G}$ и $\mathcal{G}'$ соответственно. Будем строить граф $\mathcal{G}'$ по правилу:

1) В графе $\mathcal{G}'$ между вершинами $x_i$ и $x_j$ дуга будет только в том случае, если она является частью какого-либо пути, начинающегося на вершине $x_{in}$.

2) Вес дуги $C'_l$, который представляет собой пару чисел $(\tau_{in}, \tau_{out})$, будет определяться из дуги $a_l$ с помощью описанного выше алгоритма.

Таким образом, зная начальную точку $x_{in}$ и время $t_0$, находим все дуги $a_i$, выходящие из нее. По алгоритму поиска оптимального времени прохождения дуги определяем веса дуг $C'_i$. Из них находим минимальные времена $\tau_i$ попадания во все смежные с $x_{in}$ вершины и номера маршрутов на которых эти минимумы

достигаются. При этом для каждой из этих вершин, $x_{in}$ записывается в качестве предшествующей вершины. Далее для всех соседей находятся все смежные вершины, за исключением предшествующей, и в качестве $t_0$ берется значение $\tau_i$. Если на каком-либо шаге текущее минимальное время в некоторой вершине $x_j$ меньше ранее вычисленного, то все, что было найдено из этой вершины ранее, становится неверным, а в качестве минимума берется текущее время, соответственно, изменяются предшествующая вершина и номер маршрута, на котором этот минимум был достигнут. Иначе вычисления из вершины $x_j$ прекращаются.

Такая реализация алгоритма позволяет найти кратчайшие (в смысле времени) пути между вершиной $x_{in}$ и всеми оставшимися вершинами при начальном времени $t_0$ (и как следствие – путь из $x_{in}$ в $x_{out}$).

Этот алгоритм не будет эффективным, поскольку при достаточно близких вершинах $x_{in}$ и $x_{out}$ он будет делать много лишних вычислений. Поэтому необходима его оптимизация.

Примем во внимание то обстоятельство, что если в вершине $x_{out}$ найдено время, за которое она может быть достигнута, и текущее время в некоторой вершине $x_i$ больше этого значения, то дальнейшие вычисления из этой вершины бессмысленны, поскольку в данной формулировке задачи не рассматриваются дуги с отрицательным весом.

Если же нас интересует сам путь, то по множеству предшествующих вершин его однозначно можно восстановить. Возьмем в качестве начальной вершины $x$ вершину, равную $x_{out}$. Пока значение вершины, предшествующей $x$, не будет равно нулю или $x_{in}$, то $x$ присваивается значение предшествующей вершины. Если после выполнения алгоритма значение $x$ стало равным нулю, то это говорит о том, что вершина $x_{out}$ не достижима из $x_{in}$ при начальном времени $t_0$. Иначе, записав все значения, которые принимала переменная $x$, в обратном порядке, мы получим сам путь. Поскольку для каждой вершины был не только определен минимум, но и найден номер маршрута, на котором этот минимум достигается, то можно восстановить весь путь с точностью

до номера маршрута и времени прибытия во все промежуточные вершины.

Пусть заданы те же начальные условия, только $t_0$ — время прибытия в вершину $x_{out}$. Необходимо найти оптимальное время отправление из вершины $x_{in}$. В этом случае существенных изменений в алгоритме не последует.

В алгоритме поиска оптимального прохождения дуги условия 1 и 2 заменяются на следующие:

1) $t^{(kl)} \geqslant t_{out}^{(kl)}[p]$,

2) $t^{(kl)} < t_{out}^{(kl)}[p+1]$,

$t_{in}^{(kl)}$ и $t_{out}^{(kl)}$ меняются ролями, а когда будут найдены $\tau^{(kl)}$, то вместо минимума по $k$ берется максимум.

В алгоритме поиска оптимального пути за начальную точку берется $x_{out}$; вместо дуг, выходящих из начальной вершины, ищутся дуги, входящие в эту вершину, минимум заменяется на максимум, а в проверке на прекращение вычислений знак неравенства меняется на противоположный. Если в вершине $x_{in}$ найдено время, за которое она может быть достигнута, и текущее время в некоторой вершине $x_i$ меньше этого значения, то дальнейшие вычисления из этой вершины бессмысленны.

Если в такой формулировке задачи нас интересует сам путь, то за начальную вершину берется значение $x$, равное $x_{in}$, условие выхода – сравнение $x$ с $x_{out}$ и нулем. А последовательность тех значений, которые принимал $x$, записывается в прямом порядке.

### *2.3. МОДИФИКАЦИЯ ПОИСКА ОПТИМАЛЬНОГО ПУТИ*

Пусть заданы граф $\mathcal{G}$ (такой же, как и в предыдущем алгоритме), начальное время $t_0$, вершина отправления $x_{in}$ и вершина прибытия $x_{out}$. Проведем преобразование исходного графа.

Граф транспортной сети $\mathcal{G}$ содержит достаточно много вершин, которым инцидентно ровно два ребра. Будем заменять ориентированную цепь, содержащую такие вершины, одной дугой, для которой начальная и конечная вершины совпадают с начальной и конечной вершинами цепи. Вес дуги будет равен суммарной

стоимости всех дуг, входящих в цепь. Далее действуем по вышеописанному алгоритму.

Построенные алгоритмы позволяют найти оптимальный по времени путь. Но возникает проблема иллюстрации результатов работы этих алгоритмов. Для визуального представления найденных маршрутов и различных объектов на карте города был применен метод, основанный на построении двумерных интерполяционных сплайнов. Идея метода состоит в том, что для отрисовки пути необходимо иметь только набор узловых точек, лежащих вдоль движения маршрута, через которые будет проходить сплайн.

## 3. Интерполяционный кубический сплайн дефекта 1

**Определение.** *Кубическим сплайном дефекта 1, интерполирующим на отрезке* $[a,b]$ *данную функцию* $f(x)$*, называется функция*

$$(1)\, g(x) := g_k(x) := a_k + b_k(x-x_k) + c_k(x-x_k)^2 + d_k(x-x_k)^3, \quad \text{при } x \in [x_{k-1}, x_k]_{k=1}^n,$$

*удовлетворяющая совокупности условий:*

1) $g(x_k) = f_k$ *(условие интерполяции в узлах сплайна);*

2) $g(x_k) \in C^2_{[a,b]}$ *(двойная непрерывная дифференцируемость);*

3) $g''(a) = g''(b) = 0$ *(краевые условия)* [1].

Для такого сплайна существую известные формулы для вычисления его коэффициентов:

$$f(x_{k-1}, x_k) = \frac{f_k - f_{k-1}}{x_k - x_{k-1}}; \tag{2}$$

$$a_k = f_k; \tag{3}$$

$$d_k = \frac{c_k - c_{k-1}}{3h_k}; \tag{4}$$

$$b_k = f(x_{k-1}; x_k) + \frac{2}{3}h_k c_k + \frac{1}{3}h_k c_{k-1}; \tag{5}$$

$$\begin{gathered} h_{k-1}c_{k-2} + 2(h_{k-1} + h_k)c_{k-1} + h_k c_k = \\ = 3f(x_{k-1}; x_k) - 3f(x_{k-2}; x_{k-1}). \end{gathered} \tag{6}$$

## 4. Построение двумерных сплайнов

### 4.1. ДВУМЕРНЫЙ НЕЗАМКНУТЫЙ КУБИЧЕСКИЙ СПЛАЙН ДЕФЕКТА 1

Пусть задан упорядоченный набор точек на плоскости

$$P_1 = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}, P_2 = \begin{pmatrix} x_2 \\ y_2 \end{pmatrix}, \ldots, P_n = \begin{pmatrix} x_n \\ y_n \end{pmatrix}. \tag{7}$$

Построим для этого набора непрерывную функцию $S(t)$, которая проходит через эти точки и имеет непрерывные первую и вторую производные.

Для удобства возьмем $t_1 := 0$, а $t_n = T$, где $T = \sum\limits_{i=1}^{n-1} \sqrt{(x_{n+1} - x_n)^2 + (y_{n+1} - y_n)^2}$. Тогда так же, как и для классического кубического сплайна дефекта 1, функция $S(t)$ имеет вид

$$\begin{gathered} S(t) := S_k(t) := A_k + B_k(t - t_k) + C_k(t - t_k)^2 + D_k(t - t_k)^3, \\ \text{при } t \in [t_{k-1}, t_k]_{k=1}^n. \end{gathered} \tag{8}$$

и удовлетворяет условиям:

а) $S(x_k) = P_k$;

б) $S(x_k) \in C^2_{[0,T]}$;

в) $S''(0) = S''(T) = 0$.

Стоит заметить, что формула (8) имеет тот же вид, что и формула (2), с той лишь разницей, что функция $S$ есть вектор-функция скалярного аргумента. То есть $S(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix}$. Учитывая этот факт, уравнение (8) можно записать следующим образом:

$$\begin{cases} x(t) := x_k(t) := a_{1k} + b_{1k}(t - t_k) + c_{1k}(t - t_k)^2 + d_{1k}(t - t_k)^3, \\ y(t) := y_k(t) := a_{2k} + b_{2k}(t - t_k) + c_{2k}(t - t_k)^2 + d_{2k}(t - t_k)^3, \\ \text{при } t \in [t_{k-1}, t_k]_{k=1}^n, \end{cases} \tag{9}$$

где $a_{ik}$, $b_{ik}$, $c_{ik}$, $d_{ik}$, $i = 1, 2$, – компоненты соответствующих векторов.

Таким образом, исходную задачу можно свести к решению методом прогонки двух систем, организуя счет по формулам (2)–(6).

### *4.2. ДВУМЕРНЫЙ ЗАМКНУТЫЙ КУБИЧЕСКИЙ СПЛАЙН ДЕФЕКТА 1*

Пусть задан упорядоченный набор точек (7). Построим для этого набора замкнутую кривую $S(t)$, которая также имеет непрерывные первую и вторую производные.

Для удобства введем: $P_{n+1} := P_0$, $S_{n+1} = S_0$, $t_0 := 0$ и $t_{n+1} := T$, где $T = \sum\limits_{i=0}^{n} \sqrt{(x_{n+1} - x_n)^2 + (y_{n+1} - y_n)^2}$. Тогда функция $S(t)$ имеет вид

$$S(t) := S_i(t) := A_i + B_i(t-t_i) + C_i(t-t_i)^2 + D_i(t-t_i)^3, \quad \text{при } t \in [t_i, t_{i+1}]_{k=0}^{n}. \tag{10}$$

и удовлетворяет условиям:

а) $S(x_k) = P_k$;

б) $S(x_k) \in C^2_{[0,T]}$;

в) $S'(0) = S'(T)$;

г) $S''(0) = S''(T)$.

Так как функция $S(t)$ должна иметь непрерывные первую и вторую производные и проходить через узлы интерполяции, то

$$\begin{cases} S_i(t_i) = A_i = P_i, \\ S_i(t_{i+1}) = S_{i+1}(t_{i+1}), \\ S_i'(t_{i+1}) = S_{i+1}'(t_{i+1}), \\ S_i''(t_{i+1}) = S_{i+1}''(t_{i+1}), \end{cases} \quad i = 0, 1, \ldots, n, \tag{11}$$

из формул (10) и (11) получаем

$$\begin{cases} P_i + B_i(t_{i+1} - t_i) + C_i(t_{i+1} - t_i)^2 + D_i(t_{i+1} - t_i)^3 = A_{i+1}, \\ B_i + 2C_i(t_{i+1} - t_i) + 3D_i(t_{i+1} - t_i)^2 = B_{i+1}, \\ 2C_i + 6D_i(t_{i+1} - t_i) = 2C_{i+1}. \end{cases} \tag{12}$$

В дальнейшем, из последней системы будем выражать все неизвестные коэффициенты через $C_i$.

Выражая из третьего уравнения системы (12) $D_i$ и учитывая,

что $S_{n+1} = S_0$, получаем

$$(13)\qquad \begin{cases} D_i = \dfrac{C_{i+1} - C_i}{3(t_{i+1} - t_i)}, \quad i = 0, 1, \ldots, n-1, \\ D_n = \dfrac{C_0 - C_n}{3(t_{n+1} - t_n)}. \end{cases}$$

Из первого уравнения системы (12) получаем

$$(14)\qquad \begin{cases} B_i = \dfrac{P_{i+1} - P_i}{t_{i+1} - t_i} - C_i(t_{i+1} - t_i) - D_i(t_{i+1} - t_i)^2, \\ \qquad\qquad i = 0, 1, \ldots, n-1, \\ B_n = \dfrac{P_0 - P_n}{t_{n+1} - t_n} - C_n(t_{n+1} - t_n) - D_n(t_{n+1} - t_n)^2. \end{cases}$$

Подставляем $D_i$ из (13) в формулу (14)

$$(15)\qquad \begin{cases} B_i = \dfrac{P_{i+1} - P_i}{t_{i+1} - t_i} - \dfrac{2}{3}C_i(t_{i+1} - t_i) - \dfrac{1}{3}C_{i+1}(t_{i+1} - t_i), \\ \qquad\qquad i = 0, 1, \ldots, n-1, \\ B_n = \dfrac{P_0 - P_n}{t_{n+1} - t_n} - \dfrac{2}{3}C_n(t_{n+1} - t_n) - \dfrac{1}{3}C_0(t_{n+1} - t_n) \end{cases}$$

Из формул (12), (13) и (15) получаем

$$(16)\qquad \begin{cases} C_i(t_{i+1} - t_i) + 2C_{i+1}(t_{i+2} - t_i) + C_{i+2}(t_{i+2} - t_{i+1}) = \\ = 3(P(t_{i+1}; t_{i+2}) - P(t_i; t_{i+1})),\ i = 0, \ldots, n-2, \\ C_{n-1}(t_n - t_{n-1}) + 2C_n(t_0 - t_{n-1}) + C_0(t_0 - t_n) = \\ = 3(P(t_n; t_0) - P(t_{n-1}; t_n)), \\ C_n(t_0 - t_n) + 2C_n(t_1 - t_n) + C_0(t_1 - t_0) = \\ = 3(P(t_0; t_1) - P(t_n; t_0)), \end{cases}$$

где

$$P(t_i; t_{i+1}) = \frac{P_{i+1} - P_i}{t_{i+1} - t_i}.$$

Стоит заметить, что система (16) не является трехдиагональной, в следствие чего ее приходится решать методом Гаусса, но поскольку матрица системы имеет специфический вид, то прямой ход метода Гаусса необходимо применять лишь для двух последних строк.

### *Заключение*

Поставленная задача имеет широкое применение. Алгоритмы для решения этой задачи могут применяться не только для поиска кратчайшего пути на графе транспортных маршрутов, но и для задачи нахождения минимального по времени пути с учетом пробок, для задачи наискорейшей рассылки сообщений в сети, для планирования производства (оптимального времени завершения проекта).

Стоит заметить, что приведенные алгоритмы будут работать, если граф $\mathcal{G}$ — стохастическая модель движения общественного транспорта. То есть вес дуги будет зависеть не только от времени и управления, но и от некоторой случайной величины $\xi$, задающей отклонение от среднего времени прохождения дуги.

Если ведется статистика, то можно определить плотность распределения $p_\xi(x)$ случайной величины $\xi$.

Если статистических данных нет, то предполагаем, что все $\xi_i \in B(\alpha, \beta)$ ($\beta$-распределение). Определяя математическое ожидание $M\xi = \dfrac{\alpha}{\alpha + \beta}$, мы преобразуем граф $\mathcal{G}$, который является стохастической моделью движения общественного транспорта, в граф $\mathcal{G}'$ — граф с детерминированными весами. Далее, находя кратчайшее расстояние между любыми двумя вершинами графа $\mathcal{G}'$ по описанному выше алгоритму, мы найдем наиболее вероятный наименьший по времени путь между теми же вершинами графа $\mathcal{G}$.

### *Литература*

1. ВЕРЖБИЦКИЙ В. М. *Основы численных методов.* – М.: Высш. шк., 2002. – 840 с.
2. КРИСТОФИДЕС Н. *Теория графов. Алгоритмический подход.* – М.: Мир, 1978. – 432 с.
3. МАЙНИКА Э. *Алгоритмы оптимизации на сетях и графах.* – М.: Мир, 1978. – 324 с.

# SOME ALGORITHMS OF OPTIMIZATION AND VISUALIZATION OF TRAFFIC FLOWS

**Taras Gordiychuk**, Izhevsk state Technical University, student (tsimplex@gmail.com).
**Alexander Itskov**, Izhevsk state Technical University, Izhevsk, Cand.Sc., assistant professor (pmi.istu@gmail.com).

*Abstract: The algorithm is given of optimal path search in a weighted graph with variable weights. The formula are derived for two-dimensional cubic splines with defect 1.*

Keywords: weighted graph, shortest path, cubic spline.

*Статья представлена к публикации членом редакционной коллегии О. П. Кузнецовым*