Например, доходы от страховой деятельности включают: заработанные страховые платежи по договорам страхования и перестрахования; комиссионные вознаграждения за перестрахование; доли от страховых сумм и страховых возмещений, уплаченных перестраховщиками; возвращенные суммы из централизованных страховых резервов; возвращенные суммы технических резервов. Обработка таких документов приносит СК доход.

Расходы СК несет из-за: выплаты страховых сумм и страховых возмещений; отчисления в централизованные страховые резервные фонды; отчислений в технические резервы, отличные от резервов незаработанных премий, в случаях и на условиях, предусмотренных актами действующего законодательства. Следует заметить, что именно обработка соответствующих документов приносит СК расходы.

Для рассматриваемой модели сформулирована оптимизационная задача, связанная с максимизацией доходов сети в целом (по числу линий обслуживания):

$$\begin{cases} W(T, m_1, ..., m_n) = \dfrac{1}{T - t_0} \displaystyle\int_{t_0}^{T} \sum_{i=1}^{n} (v_i(t) - d_i N_i(t) - E_i m_i) dt \to \max\limits_{m_1, m_2, ..., m_n}, \\ m_i \le a_i,\ i = \overline{1, n}, \end{cases}$$

где E_i – стоимость содержания одной линии обслуживания, занимающейся обработкой заявок в i-й СМО, а d_i – плата за пребывание (задержку) одной заявки в i-ой СМО (в очереди и на обслуживании), $v_i(t)$ – ожидаемые доходы i-й СМО, $i = \overline{1, n}$. Методика нахождения ожидаемых доходов $v_i(t)$, $i = \overline{1, n}$, описана в [2].

Получили задачу условной целочисленной оптимизации. Она решалась методом полного перебора. Решением задачи является число сотрудников организации, при котором целевая функция (в нашем случае доход организации) будет максимальным.

Список цитированных источников

1. Баканова, Н.Б. Моделирование процесса движения документов в корпоративных системах документооборота / Н.Б. Баканова, В.М. Вишневский // Автоматика и телемеханика – 2008. – №9. – С. 183 – 188.

2. Маталыцкий, М.А. Системы и сети массового обслуживания: анализ и применение: монография / М.А. Маталыцкий, О.М. Тихоненко, Е.В. Колузаева. – Гродно: ГрГУ, 2011. – 817 с.

УДК 656.2-50: 519.8

ВСТРЕЧНЫЙ ПОИСК КРАТЧАЙШИХ МАРШРУТОВ НА СЕТЯХ С ПРЕДОПРЕДЕЛЕННЫМИ РЕШЕНИЯМИ

Кароли М.К.

Белорусский государственный университет информатики и радиоэлектроники, г. Минск

Научный руководитель: Ревотюк М.П., к.т.н., доцент

Известно, что при поиске кратчайших путей на нагруженном ориентированном графе $G(N, A)$, где N – множество вершин, A – множество дуг, время построения дерева путей (поиска решения) растет квадратично или, по меньшей мере, при тщательно построенной вычислительной схеме по закону $x \cdot \log_2 x$ с увеличением расстояния x от корня дерева до целевой вершины [1]. Классический алгоритм Дийкстры с отображением очередей на Фибоначчиевы кучи характеризуется сложностью $O(m + n \cdot \log_2 n)$, где $m = |A|$, $n = |N|$.

Известные приемы ускорения процесса поиска кратчайших путей, такие как целенаправленный поиск, встречный поиск, многоуровневый подход [2], базируются на ограничении локальных областей поиска. Далее будем полагать, что построение дерева путей идет по волновой схеме однократного просмотра дуг, реализуемой алгоритмом Дийкстры, а граф транспортной сети представлен структурой смежности:

$$FSF = \left\{S_i = \left\{(j, w_{ij}) : (i, j) \in A\right\},\ i \in N\right\},\ w_{ij} \to R^+,\ (i, j) \in A. \qquad (1)$$

Состояние поиска решения алгоритмом Дийкстры представляется массивом расстояний $D = \{D_i, i \in N\}$, а также очередью вершин, элементы которой упорядочены по значению расстояния от корня дерева.

Можно заметить, что в процессе развития дерева кратчайших путей из заданной вершины каждая вершина окончательного дерева как минимум один раз будет представлена в очереди вершин. Однако некоторые вершины в очереди побывают один раз, если для них первоначально выбраны входные дуги минимальной длины [1]. Такие дуги можно выделить до начала поиска. Эффективный прием фильтрации просматриваемых дуг – каждой дуге поставить в соответствие список вершин, кратчайшие пути к которым включают такую дугу [2]. Построение подобных списков возможно после предварительного построения всех деревьев кратчайших путей. В результате процесс поиска дерева путей идет на графе, дуги которого ассоциированы с целевой вершиной.

Достаточно часто на практике встречаются задачи поиска маршрута между двумя заданными вершинами сети. В этом случае целесообразно организовать процесс поиска путем построения двух встречно растущих деревьев. В результате объем анализируемых данных сокращается в два раза. Дерево из конечной вершины должно строиться на графе с обратным направлением дуг, поэтому представление модели сети задается расширенным графом – объединением (1) и инверсии (1). Предлагается учесть ассоциации дуг с целевыми вершинами характеристическими множествами признаков вхождения вершин в заранее выделенные подмножества вершин.

Встречный поиск принципиально можно реализовать любой процедурой построения дерева от одной вершины до всех остальных на расширенном графе, отражающем фактически существующее пространство поиска. Оптимальный размер деревьев поиска, как легко показать, соответствует одинаковому расстоянию от корней дерева.

Предлагается улучшить известные процедуры встречного поиска посредством остановки процесса после выявления предопределенных решений

$$T_j = \left\{(i, j) : i = \arg\min_{i,j}\left\{w(i, j) : (i, j) \in A\right\}\right\},\ j \in N. \qquad (2)$$

Обозначим исходный граф, заданный в виде (1), через $G^+(N^+, A^+)$, а граф с инвертированием направления дуг – $G^*(N^*, A^*)$. Множества дуг таких графов могут не включать дуги без ассоциаций с начальной и конечными вершинами пути. С целью удобства организации процесса ветвления желательно использовать общую очередь, для чего множество вершин и дуг графа $G^*(N^*, A^*)$ определим так:

$$\begin{cases} N^* = \left\{x^* = x^+ + n,\ x^+ \in N^+\right\}, \\ A^* = \left\{(x^*, y^*) = (y^+ + n,\ x^+ + n),\ (x^+, y^+) \in A^+\right\}. \end{cases} \qquad (3)$$

Вершины x^+ и x^* будем далее называть сопряженными. Легко заметить, что условие отбора предопределенных решений для графа $G^*(N^*, A^*)$ имеет вид

$$T_j^* = \left\{(i,j) : j = \arg\min_{i,j}\{w_{ij} : (i,j) \in A^+\}\right\},\ j \in N^+.$$

В общем случае связь сопряженных вершин пусть задается функцией

$$conj(x) = x^* \cdot (x \in G^+) + x^+ \cdot (x \in G^*),\ x \in N^+ \cup N^*.$$

В случае же нумерации вершин по правилу (3)

$$conj(x) = (x+n) \cdot (x < n) + (x-n) \cdot (x \ge n)\ x \in N^+ \cup N^*.$$

Пусть заданы *s* и *f* – начальная и конечная вершины исходного графа G^+. Так как, по определению (3), $N^+ \cap N^* = \varnothing$, то построение деревьев можно проводить параллельно, организовав синхронное движение волны от корней дерева на графе

$$G(N,A) = G\left(N^+, A^+\right) \cup G\left(N^*, A^*\right).$$

Для этого достаточно начать процесс ветвления из вершин $s \in G^+$ и $f^* = conj(f)$, $f^* \in G^*$. В отличие от известного приема поочередного развития деревьев, синхронное движение оказывается оптимальным для “географических” графов.

Организация встречного поиска требует определения правила остановки. Можно показать, что остановка должна соответствовать моменту фиксации постоянной пометки вершины дерева, когда сопряженная вершина уже является постоянно помеченной.

Если для некоторого дерева кратчайших маршрутов максимальное расстояние от постоянно помеченных вершин до корня есть *d*, то признаком постоянной пометки вершины *x* является условие $D_x \le d$. В рассматриваемом случае для обоих деревьев значение *d* одинаково.

Отсюда следует, что правило остановки можно определить на значениях текущих расстояний – $D_{conj(i)} \le D_i$, где *i* – вершина графа $G(N^+, A^+)$ или графа $G(N^*, A^*)$, получающая постоянную пометку.

Таким образом, использование синхронного движения от корней деревьев не требует хранения пометок, а момент остановки совпадает с моментами выделения критических вершин дерева маршрутов.

В случае наличия предопределенных решений правило остановки можно определить относительно фактов пометки обеих вершин выделенной дуги. Факт пометки конечной вершины дуги, имеющей постоянную пометку в инвертированном графе, – достаточное условие остановки. Правило остановки здесь: $(D(y) < \infty) \wedge ((x,y) \in T_j),\ y = conj(x)$. Очевидно, что проверка такого предиката в задаче поиска кратчайших путей на классическом нагруженном графе может стать громоздкой.

После остановки в вершине *x* остается достроить маршрут до конечной вершины в исходном графе. Так как остановка может быть обнаружена в любом из встречно растущих деревьев, а результат поиска необходимо получить лишь для дерева из исходной вершины, то для перехода в такое дерево используем функцию

$$orig(x) = x \cdot (x \in G^+) + conj(x) \cdot (x \in G^*),\ x \in N^+ \cup N^*.$$

В случае нумерации вершин расширенного графа по правилу (3)

$$orig(x) = x \cdot (x < n) + (x-n) \cdot (x \ge n),\ x \in N^+ \cup N^*.$$

Таким образом, построенные процедуры поиска используют для представления модели сети память в два раза увеличенного объема. Структура представления модели сети остается эффективнее матричных моделей, по крайней мере, с точки зрения удобства коррекции описания сети в реальном времени.

Проведенные рассуждения базировались на предположении, что состояние процесса поиска отражается на очереди вершин – листьев текущего дерева. В случае наличия ограничений на пути на графе [1] приходится использовать очередь дуг. Однако легко заметить, что и здесь помеченные дуги (2) по тем же соображениям являются пригодными для ускоренного включения в дерево кратчайших путей. Правило остановки остается без изменений.

Эксперименты показывают, что на графе реальной сети автомобильных дорог, где $|N| \cong 10^4$, $|A| \cong 10^5$, среднее время поиска кратчайших маршрутов между случайными парами вершин сокращается более, чем в два раза. Степень сокращения зависит от количества подмножеств вершин кратчайших путей, ассоциируемых с дугами.

Список цитированных источников

1. Ревотюк, М.П. Поглощение предопределенных решений жадными алгоритмами / М.П. Ревотюк, Н.И. Застенчик, Е.В. Шешко // Известия Белорусской инженерной академии. – № 1(17)/2, 2004. – С. 112–114.

2. Holzer, M. Combining Speed-up Techniques for Shortest-Path Computations / M. Holzer, F. Schulz, D. Wagner, T. Wilhalm // ACM Journal of Experimental Algorithmics. – Vol. 10, Article No. 2.5, 2005. – P. 1–18.

УДК 517.3

ПРИЛОЖЕНИЯ КОНФОРМНОГО ОТОБРАЖЕНИЯ

Карпук И.А.

Брестский государственный технический университет, г.Брест

Научный руководитель: Лебедь С.Ф., к.ф.-м.н., доцент

Определение. Взаимно-однозначное отображение области D на область D* называется конформным, если в окрестности любой точки области D главная линейная часть этого отображения есть ортогональное преобразование, сохраняющее ориентацию.

Из этого определения вытекают два основных свойства конформных отображений:

1. Конформное отображение преобразует бесконечно малые окружности в окружности с точностью до малых высших порядков (круговое семейство).

2. Конформное отображение сохраняет углы между кривыми в точках их пересечения (свойство сохранения углов). Предполагается, что при конформном отображении направление отсчёта углов не изменяется, т.е. сохраняется взаимное расположение (ориентация) кривых. Иногда такое отображением называют конформным отображение первого рода, тогда как при сохранении углов, но изменении ориентации кривых, говорят о конформном отображении второго рода.

Утверждение. Отображение с помощью аналитической, однолистной в конечной области *D* функции является конформным в *D*.

Если функция $\omega = f(z)$, аналитическая в *D*, осуществляет взаимно однозначное отображение, то точки ω называются образами точек *z*, а точки *z* – прообразами. В силу свойств взаимно однозначного отображения образом области *D* как открытого множества, состоящего из внутренних точек, является область *G*, а образом кривой γ – границы области D ($\gamma = \delta D$) – является кривая Γ – граница области G ($\Gamma = \delta G$).

В теории и практике конформных отображений ставятся и решаются две задачи.

Первая задача: нахождении образа данной линии или области при заданном отображении – прямая задача.

Вторая – нахождение функции, осуществляющей отображение данной линии или области на другую заданную линию или область – обратная задача.