ЗАДАЧА О БЫСТРЕЙШЕМ ПЕРЕМЕЩЕНИИ НА ГРАФЕ С ПЕРЕМЕННЫМИ ЗАДЕРЖКАМИ

И.В.Коннов

Казань, эл.почта: konn-igor@ya.ru

Для динамической задачи о быстрейшем перемещении между заданными вершинами графа с произвольными функциями задержек предложен и обоснован модифицированный алгоритм Дейкстры.

Ключевые слова: Экстремальные задачи на графах, задача о быстрейшем перемещении, переменные задержки дуг, алгоритм типа Дейкстры.

1 Постановка задачи

Напомним, что граф задается множеством вершин V, которое будет считаться конечным, множеством дуг $\mathcal{A}$ и отображением, ставящим в соответствие паре вершин i и j элемент $a \in \mathcal{A}$. Если значение отображения для i и j непусто, то определена дуга $a = (i, j)$, для которой вершина i является исходящей (начальной), а j – входящей (конечной). Если значение отображения для i и j пусто, то вершины i и j не связаны дугой. Последовательность дуг

$$a_1, a_2, \ldots, a_{k-1},$$

где $a_l = (i_l, i_{l+1})$ или (i_{l+1}, i_l) и любые две соседние дуги имеют смежную вершину, называется *цепью*, соединяющей вершины i_1 и i_k, если $i_1 \notin a_2$ и $i_k \notin a_{k-2}$. Граф называется *связным*, если любые две его вершины можно соединить цепью. *Циклом* называется замкнутая цепь, у которой начальная и конечная вершины совпадают. *Путем* называется цепь, которую можно пройти от начальной до конечной вершины в направлении дуг. Граф называется *связным*, если любые две его вершины можно соединить цепью. Граф называется *связным по путям*, если любые две его вершины можно соединить путем.

С графами связано множество прикладных экстремальных задач. Одной из наиболее известных и часто используемых является задача о кратчайшем связывающем пути в графе. Пусть каждой дуге графа $a = (i, j) \in \mathcal{A}$ приписан вес (длина) $c_a = c_{ij}$. Задача будет состоять в том, чтобы для заданного графа и пары его вершин v и w найти путь с минимальным суммарным весом дуг из v в w. Веса дуг могут иметь различный содержательный смысл. Если вес дуги $a \in \mathcal{A}$ – это время прохождения дуги (задержка), то получаем задачу о быстрейшем перемещении между заданными вершинами графа. Наиболее изученной является задача о кратчайшем пути с постоянными весами.

В настоящей работе изучается задача о быстрейшем перемещении между заданными вершинами графа с переменными задержками дуг. Каждой дуге графа $a = (i, j) \in \mathcal{A}$ приписывается неотрицательная функция $t_{ij}(d_i)$ – время прохождения дуги a, если движение из вершины i начато в момент времени d_i. Требуется для заданной пары вершин v и w и начального момента времени d'_0 найти путь с минимальным временем перемещением по дугам из v в w. Для упрощения далее считаем, что $d'_0 = 0$.

Сложность решения этой задачи зависит от применяемых дополнительных условий. В частности, можно допустить или разрешить простои в движении в вершинах, причем задача без простоев гораздо сложнее. Можно ввести дополнительные условия на функцию задержки. В частности, в [1] эта задача изучалась при условии монотонности момента прибытия по отношению к моменту начала движения, что автоматически влечет невыгодность простоев в вершинах. Кроме того, предполагалось, что $t_{ij}(d_i) < +\infty$ для всех дуг $(i, j) \in \mathcal{A}$ и всех моментов $d_i \geq 0$. Там же в этом случае предложен и обоснован простой алгоритм Дейкстры. Однако во многих приложениях функции задержки могут быть достаточно произвольными, не обладать свойствами монотонности. Тогда алгоритм Дейкстры неприменим при требовании отсутствия простоев, что иллюстрирует следующий пример.

Пример 1 Граф содержит четыре вершины, которые связаны пятью дугами, как показано на рис. 1. Пусть $v = v_1$ и $w = v_4$, $t_{v_1,v_2} \equiv 2$, $t_{v_1,v_3} \equiv 5$, $t_{v_2,v_3} \equiv 2$, $t_{v_2,v_4} \equiv 7$,

$$t_{v_3,v_4}(d_{v_3}) = \begin{cases} 4, & \text{если } 4 \leq d_{v_3} < 5, \\ 2, & \text{если } 5 \leq d_{v_3} \leq 6. \end{cases}$$

Тогда путь $(v_1 \to v_2 \to v_3 \to v_4)$ дает 8, причем подпуть $(v_1 \to v_2 \to v_3)$ – быстрейший, путь $(v_1 \to v_2 \to v_4)$ дает 9, но путь $p = (v_1 \to v_3 \to v_4)$ дает 7. Это быстрейший путь, хотя его часть $(v_1 \to v_3)$ не наилучшая.

Отметим, что произвольные функции задержки могут быть связаны не только с различными режимами работы дуг по времени, но и с использованием различных средств перемещения по дугам. Задача о быстрейшем перемещении в графе существенно усложняется в том случае, когда дополнительно $t_{ij}(d_i) = +\infty$ для

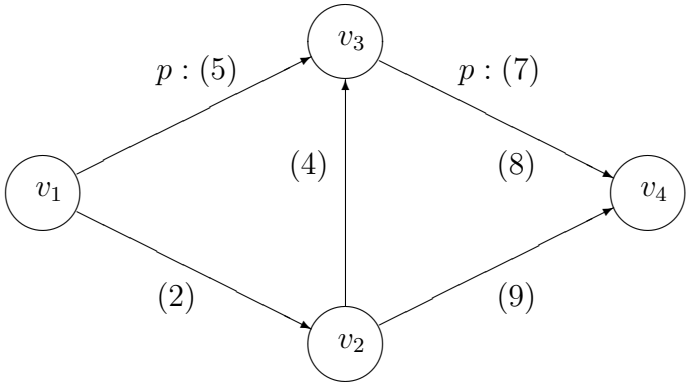

Рис. 1: Наилучший путь содержит невыгодный подпуть

некоторых дуг $(i, j) \in \mathcal{A}$ и некоторых моментов d_i. Фактически дуга (i, j) для таких периодов времени закрыта для перемещения. Тогда можно формально все дуги (i, j) включить в множество $\mathcal{A}$, а реальную топологию графа определять для каждого момента времени по открытым дугам. Граф назовем *связным по перемещениям*, если для любой пары вершин $v, w \in V$ существует последовательность вершин

$$i_1, i_2, \ldots, i_l,$$

где $i_1 = v$, $i_l = w$, и соответствующая последовательность моментов времени

$$d_1, d_2, \ldots, d_l,$$

такие что $d_1 \geq 0$, $d_s \geq d_{s-1} + t_{i_{s-1}, i_s}(d_{s-1})$, $s = \overline{2, l}$, $d_l < +\infty$.

2 Алгоритм и его обоснование

Для задачи о быстрейшем перемещении между заданными вершинами v и w графа с возможностью простоев в движении в вершинах опишем модифицированный алгоритм Дейкстры. Граф предполагается связным по перемещениям, функции задержек предполагаются произвольными неотрицательными и полунепрерывными снизу функциями, тогда $d_i + t_{ij}(d_i) \to +\infty$, если $d_i \to +\infty$.

Алгоритм. Вначале полагаем $V' = \varnothing$, $V'' = V$, $H' = \varnothing$, $t'_v = 0$, $t'_j = +\infty$ для $j \in V''\backslash\{v\}$.

На очередной итерации выбираем элемент k из V'', такой что

$$t'_k = \min_{j \in V''} t'_j,$$

полагаем $V' = V' \bigcup \{k\}$, $V'' = V''\backslash\{k\}$. Если $V' \neq \{k\}$, то также полагаем $H' = H' \bigcup \{(i(k), k)\}$. Если $k = w$, то останов. Далее для каждой вершины $s \in V''$, $(k, s) \in \mathcal{A}$ вычисляем

$$d'_{ks} = \arg\min\{d_k + t_{ks}(d_k) \mid d_k \geq t'_k\},$$

полагаем $t''_s = d'_{ks} + t_{ks}(d'_{ks})$. Если $t''_s < t'_s$, то полагаем $t'_s = t''_s$, $i(s) = k$.

В этом алгоритме величина t'_i определяет текущее наименьшее время перемещения из вершины v в i, V' определяет текущее множество вершин, для которых точное наименьшее время перемещения вычислено, V'' определяет текущее остающееся множество вершин, H' определяет дерево дуг с наименьшим временем перемещения. Если в алгоритме выполнять останов по условию $V'' = \varnothing$ вместо условия $k = w$, то алгоритм найдет быстрейшие перемещения между вершиной v и всеми остальными вершинами графа.

Теорема. Алгоритм для любого i из V' определяет точное наименьшее время перемещения t'_i между вершиной v и вершиной i, в то же время H' определяет для них дуги с наименьшим временем перемещения.

Для обоснования будет достаточно показать, что при определении индекса k на любой итерации величина t'_k определяет точное наименьшее время перемещения между вершиной v и вершиной k.

Вначале при $k = v$ это очевидно, поскольку $t'_v = 0$, $t'_j = +\infty$ для $j \in V''\backslash\{v\}$. Следующим будет выбран индекс k, такой что $(v, k) \in \mathcal{A}$ и

$$t'_k = d'_{vk} + t_{vk}(d'_{vk}) \leq t'_j = d'_{vj} + t_{vj}(d'_{vj}),\ \forall (v, j) \in \mathcal{A}.$$

Это значит, что индекс k на этой итерации определяет вершину и точное наименьшее время перемещения t'_k между вершиной v и вершиной k, поскольку остальные величины не могут быть меньше.

Предположим, что после конечного числа итераций и полученных текущих множеств V' и V'' для любого i из V' величина t'_i определяет точное наименьшее время перемещения между вершиной v и вершиной i. Пусть выбран индекс $k = i_p \in V''$ на итерации, но величина t'_k не является наименьшим временем перемещения между вершиной v и вершиной k. Это значит, что существует последовательность вершин

$$i_1, i_2, \ldots, i_p,$$

и соответствующая последовательность моментов времени

$$d''_{i_1}, d''_{i_2}, \ldots, d''_{i_p},$$

такие что $i_1 = v$, $d''_{i_1} \geq 0$, $d''_{i_s} \geq d''_{i_{s-1}} + t_{i_{s-1},i_s}(d''_{i_{s-1}})$, $s = \overline{2,p}$, а также $d''_{i_p} < t'_k$. Тогда можно определить последовательность моментов времени

$$\tilde{t}_{i_1}, \tilde{t}_{i_2}, \ldots, \tilde{t}_{i_p},$$

так чтобы $\tilde{t}_{i_1} = 0$,

$$\tilde{t}_{i_s} = \min\{d_{i_{s-1}} + t_{i_{s-1},i_s}(d_{i_{s-1}}) \mid d_{i_{s-1}} \geq \tilde{t}_{i_{s-1}}\},\ s = \overline{2,p}.$$

Отсюда следует $\tilde{t}_{i_s} \leq d''_{i_s}$, $s = \overline{1,p}$, поэтому $\tilde{t}_{i_p} < t'_{i_p}$. Таким образом, величина $\tilde{t}_{i_s}$ определяет наименьшее время перемещения между вершиной v и вершиной i_s для выбранного пути, который лучше полученного в алгоритме.

Отметим, что при этом найдется пара вершин $i_{l-1} \in V'$, $i_l \notin V'$, поскольку $v = i_1 \in V'$, $k = i_p \notin V'$, возможно, что $l = p$. Пусть l – наименьший номер вершины в пути с таким свойством. По построению пути имеем $\tilde{t}_{i_l} \leq \tilde{t}_{i_t} < t'_{i_p}$, но $i_l \notin V'$. С другой стороны, по предположению, $t'_{i_{l-1}} \leq \tilde{t}_{i_{l-1}}$, тогда $t'_{i_l} \leq \tilde{t}_{i_l}$ для итерации, где был выбран индекс $k = i_{l-1}$. Действительно, индекс i_{l-1} по алгоритму должен был попасть в V' раньше текущей итерации, тогда же было вычислено значение t'_{i_l} – текущее наименьшее время перемещения из вершины v в i_l. Поскольку в ходе итераций значение t'_{i_l} не могло увеличиться, то на текущей итерации имеем $t'_{i_l} \leq \tilde{t}_{i_l} < t'_{i_p}$, но $i_l \notin V'$. Однако это неравенство противоречит правилу выбора индекса k на итерации.

3 Обсуждение

Прежде всего заметим, что в данной постановке задача о быстрейшем перемещении в графе может и не иметь решения, если граф не является связным

по перемещениям. Предложенный алгоритм позволяет отследить эту ситуацию. Она может возникнуть, когда $t_{ij}(d_i) = +\infty$ для некоторых моментов d_i, и дуга (i, j) закрыта для перемещения для каких-то периодов времени. Такая ситуация достаточно типична для многих реальных транспортных сетей и сетей связи. Требование полунепрерывности снизу функций задержек тогда влечет замкнутость подмножества полуоси, где дуга открыта для перемещения. Это может быть, например, циклический набор отрезков.

Постановка задачи с закрытыми периодами времени дуг представляет интерес и в случае постоянных весов открытых дуг (времен перемещения), т.е. когда $t_{ij}(d_i) \equiv l_{ij}$ для открытых периодов времени дуги. Если же $l_{ij} = 0$, то это просто задача о наименьшем времени ожидания при перемещении между вершинами v и w. В общем случае гарантировать существование решения задачи о быстрейшем перемещении можно, например, когда для любой дуги $(i, j) \in \mathcal{A}$ и любого момента $d_i \geq 0$ найдется число $\tilde{d}_i > d_i$, такое что $t_{ij}(\tilde{d}_i) < +\infty$.

Хорошо известно, что существенные колебания параметров потока достаточно типичны для различных транспортных сетей и сетей связи. Поэтому постановка задачи о быстрейшем перемещении с произвольными функциями задержки гораздо лучше будет подходить в качестве математической модели реальной сети. Более того, такая постановка удобна даже при стационарном поведении транспортного потока, когда перемещение по дуге может происходить в несколько выбранных дискретных моментов времени, по расписанию. Если все транспортные средства на дуге однотипны, то в функцию задержки можно включить и время ожидания рейса.

Пример 2 Пусть в промежутке времени $[0, 4]$ по дуге (i, j) выполняются два рейса по 1 часу: в 1 час и в 4 часа. Тогда можно определить функцию задержки

$$t_{ij}(d_i) = \begin{cases} (1 - d_i) + 1, & \text{если } 0 \leq d_i \leq 1, \\ (4 - d_i) + 1, & \text{если } 1 < d_i \leq 4. \end{cases}$$

Здесь движение отождествляется с выбором рейса.

Кроме того, по одной дуге в разное время могут выполняться рейсы разными транспортными средствами, с разным временем рейса. Тогда удобнее отметить все дискретные моменты времени начала рейсов с указанием реального времени задержки, а затем построить интерполяционную кусочно-линейную функцию задержки для этой дуги.

Такой подход позволяет учитывать также другие факторы при выборе маршрута перемещения, например, стоимость и надежность. В частности, при установке ограничений сверху на стоимость и снизу на надежность при перемещении по дуге следует исключить не удовлетворяющие им транспортные средства и скорректировать функцию задержки. Аналогичным образом можно оценивать по стоимости и надежности сами дуги в целом. С другой стороны, после получения наилучшего решения можно искать чуть худшие по времени, но зато лучшие по другим

критериям маршруты, сделав соответствующую корректировку алгоритма. В силу небольшой сложности алгоритма по числу итераций он может применяться и при необходимости модификации маршрута перемещения из-за изменения свойств графа в режиме реального времени.

Решение задачи выбора шага по времени d'_{ks} зависит от свойств функции t_{ks}. Если эти функции дифференцируемы, то можно проверять корни уравнения $t'_{ks}(d_k) = -1$ при $d_k > t'_k$, а также $d_k = t'_k$. Требование полунепрерывности снизу функций задержек можно снять, достаточно, чтобы задача выбора числа d'_{ks} на каждой итерации имела решение.

Список литературы

[1] Гимади Э.Х., Глебов Н.И. Дискретные экстремальные задачи принятия решений. – Новосибирск: НГУ, 1991.