МОНИТОРИНГ ИЗМЕНЕНИЯ РЕЗУЛЬТАТОВ ОПТИМИЗАЦИИ ПУТЕЙ НА ГРАФАХ ТРАНСПОРТНЫХ СЕТЕЙ

М.П. РЕВОТЮК, Н.В. ХАДЖИНОВА

Белорусский государственный университет информатики и радиоэлектроники, П. Бровки, 6, Минск, 220013, Республика Беларусь

Аннотация. Рассматривается процесс слежения за изменением результатов оптимизации путей на графах транспортных сетей после получения информации об изменении структуры сети, ее параметров или ограничений в реальном времени. Предложены открытые для расширения структура данных и алгоритм реоптимизации кратчайших путей в лесах подграфов графа транспортной сети, снижающие вычислительную сложность обновления решений пропорционально объему поступивших данных.

Ключевые слова: транспортные сети, кратчайшие пути, метод реоптимизации, мониторинг изменений.

Abstract. The process of monitoring the changes in the optimization of paths on the graphs of transport networks results after receiving information about changes in the network structure, its parameters or limitations in real time is considered. The opened for expansion data structure and an algorithm for reoptimizing the shortest paths in the forests of the subgraphs of the transport network graph are proposed, which reduce the computational complexity of updating solutions in linear proportion to the amount of received data.

Keywords: transport network, shortest paths, reoptimization method, monitoring of changes.

Введение

При решении задач оптимизации перемещений и перевозок на транспортных сетях нет полной информации о наличии изменений структуры и параметров сети в будущем, а также об эффективности или приемлемости принятых решений. Важнейшая в транспортных системах задача поиска кратчайших путей на графах может потребовать в практически интересных случаях уточнения весов дуг графа из-за появления транспортных пробок, ремонтных работ и различных ограничений. Характерная особенность транспортных операций – независимость и асинхронность появления возмущений оптимальных решений, а также отсутствие глобального влияния отдельных возмущений на множество всех процессов на сети. Тем самым очевидно определяется целесообразность реализации идеи реоптимизации существующих решений в реальном времени по мере поступления информации о возмущениях.

Постановка задачи

Известно, что для поиска кратчайших путей на нагруженном ориентированном графе $G(N,A)$, где N – множество вершин, A – множество дуг с весовой функцией $W: A \to R^+$, лучшим является алгоритм Дейкстры [1]. Процесс построения дерева кратчайших путей имеет волновой характер однократного просмотра дуг до исчерпания возможности его развития из исходной вершины и характеризуется сложностью $O(m + n \cdot \log_2 n)$, где $m = |A|, n = |N|$.

Очевидно, что реализация алгоритма Дейкстры на графах с изменяемой структурой порождает проблему выбора способа представления графа и пространства состояний процесса поиска решения. Нагруженный граф $G(N,A)$, представляющий транспортную сеть, обычно имеет незначительную степень связности вершин, поэтому практически задается списком дуг и полностью определяется тернарным отношением $G(x,y,w)$, где x, y – концевые вершины дуги с весом w. В таких случаях описание графа задано легко модифицируемым списком дуг с расширяемыми множествами N и A вершин.

Задач поиска кратчайших путей для координируемых операций на транспортных сетях объективно локальны. Такие задачи можно представить отношением $F(a,o,d)$, где a – агент, нуждающийся в описании кратчайшего пути из вершины o в вершину d графа сети, $o,d \in N$. В любой момент времени существует потребность наличия информации о кратчайших путях из множества $X = \{x \mid (?,x,?) \in F$ до множества $Y = \{y \mid (?,?,y) \in F$. На практике для отдельной транспортной системы всегда справедливо $|X| << |N|$ и $|Y| << |N|$, а информация о кратчайших путях актуальна лишь для множества агентов $Z = \{z \mid (z,?,?) \in F$.

В установившемся состоянии сети такая информация может выбираться из множества леса деревьев кратчайших путей, сохраняемых в таблице отношения $T(s,x,p,d)$, где: s – корневой узел дерева, $s \in \overline{\overline{X}}$; x – узел дерева, $x \in N$; d – расстояние от корня s до узла x, p – номер предшествующего узла кратчайшего пути в узел x. Здесь $\overline{\overline{X}}$ – потенциально интересный для конкретной транспортной системы набор корневых вершин дерева из леса, $\overline{\overline{X}} \subseteq N$. Кроме этого, полезно определить ограничение глубины деревьев кратчайших путей множеством конечных вершин $\overline{\overline{F}}$, когда всегда справедливо $Y \subseteq \overline{\overline{F}}$.

Информация о возмущениях транспортной сети может быть задана списком дуг с изменившимися весами, представленным отношением $R(x,y,w)$, где x, y – концевые

вершины дуги с весом w (значение $w=\infty$ означает запрос удаления дуги $x \to y$). При этом не обязательно выполнение условия $x, y \in N$, что означает отсутствие запрета на коррекцию любой части графа транспортной сети. Если в этот момент выполняется условие $(x \in X) \vee (y \in Y)$, то множество агентов $J = \{a \mid (a,o,d) \in T(o,d,?,?)\}$ должно получить информацию о новом пути после его реоптимизации.

Таким образом, фазы перехода между устанавливающимися состояниями процесса слежения за оптимальностью построенных на сети путей включают выявление факта $R \neq \varnothing$, этап реоптимизации и перестройки $T(s,x,p,d)$ для изменившегося отношения $G(x,y,w)$, выявление множества заинтересованных в актуальности информации агентов J и передачу им описания изменения оптимального пути.

Предмет обсуждения – реализация процесса мониторинга состояния транспортной сети и передачи результатов решения задачи реоптимизации дерева путей. В отличие от известных постановок задач реоптимизации дерева путей [1-4], здесь будут учитываться интересы потребителей результатов решения отдельных локальных задач в лесу подграфов исходного графа. Для этого, кроме задержек времени на перестройку дерева путей, необходимо учесть процесс формирования и передачи актуальных уведомлений об изменении кратчайших путей.

Модель графа и пространства состояния поиска

Реляционная модель графа и пространства поиска решения – основа эффективной по памяти предлагаемой далее схемы поиска дерева путей методом бутстрэппинга на основе аналогии складывания ветвей дерева леса от корня до последнего постоянно включаемого в дерево узла. Такой метод отличается возможностью реализации на реляционных базах данных, что обеспечивает открытость расширения модели для учета различных ограничений.

Рассмотрим вначале форму задания графа, а затем модель переменных состояния поиска кратчайших путей. Здесь будут использованы понятия отношения и представлений (view) таблиц в терминах реляционных баз данных.

Пространство состояний поиска решения в традиционных реализациях алгоритма Дейкстры соответствует графу всей транспортной сети и включает D – массив расстояний от корня дерева, P – массив номеров предшествующих вершин, Q – очередь вершин, где элементы упорядочены по текущему значению расстояния от корня дерева [1,2]. При этом $|D| = |N|$, $|P| = |N|$, $|Q| \leq |N|$.

Метод бутстрэппинга всегда работает с подграфом исходного графа. В случае решения задач поиска кратчайших путей на множестве вершин некоторой компактной части графа возможна несложная модификация алгоритма Дейкстры, когда поиск проводится на расширяемом подграфе с последовательно включаемыми вершинами создаваемого дерева кратчайших путей от заданной корневой вершины (рис. 1).

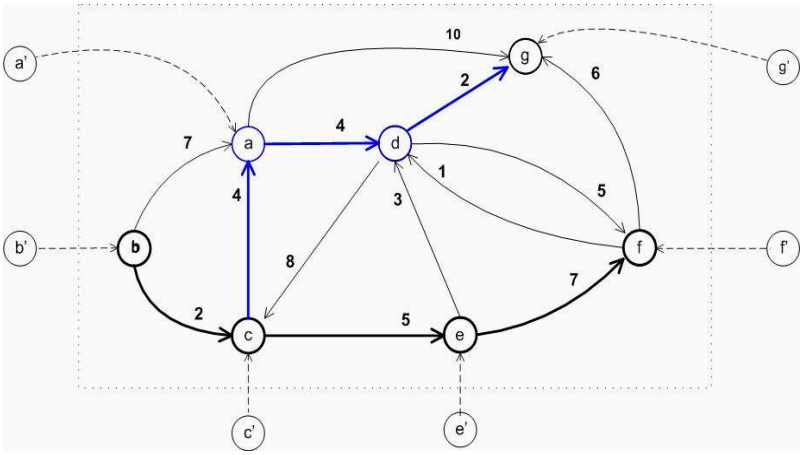

Рис. 1. Пример задачи поиска кратчайших путей на подграфе

Здесь жирными линиями выделены дуги дерева кратчайших путей с корнем в вершине $s = b,$ среди которых черным цветом выделены дуги кратчайшего пути $s \to f$.

Для реализации алгоритма Дейкстры для каждой вершины x требуется представление множества непосредственно достижимых смежных вершин $x', x' = \{k \mid w(x,k) > 0\},$ где $w(x,k)$ – вес дуги $x \to k,\ x,k \in N.$ На основе таблицы отношения $G(x,y,w)$ легко построить его виртуальное представление $G_x(x,y,w)$ в форме FSF (Forward Star Form). Индексации кортежей такой таблицы по ключу x формально определяет множество связей $\{(x,y,w),\ y \in x'\},\ x \in N.$ После этого проверка условия $x \in N$ состоит в оценке успеха поиска кортежа с ключом x в таблице G_x.

Обозначим для каждой вершины y множество входных смежных вершин $'y,$ $'y = \{k \mid w(k,y) > 0\},$ где $k, y \in N.$ Виртуальное представление $G_y(x,y,w)$ в форме BSF (Backward Star Form) инверсии графа образуется после индексации таблицы отношения *Graph* по ключу y. Такое представление определяет множество кортежей $\{(x,y,w),\ x \in 'y\},\ y \in N.$ Успех поиска кортежа с ключом y в таблице G_y означает истинность условия $y \in N$.

Таким образом, имеем $N = N_x \bigcup N_y,$ что означает отсутствие необходимости явного отдельного определения множества $N.$ Операция расширения графа сети $G(N,A)$ реализуется на логическом уровне модели данных без использования понятия адресной функции. Это влечет отсутствие проблем комплексирования существующих моделей транспортной сети из разных источников и разной степени детализации. Кроме этого, доступно на высоком описание ограничений на условия перемещения на сети в терминах функций как отношений.

Дерево кратчайших путей или любое его поддерево – связный граф по определению. Если s – исходная вершина, а x – произвольный узел или лист дерева путей, $s, x \in N,$ то текущий или кратчайший путь $s \to x$ можно восстановить обратным движением из листа x:

$$p(x,s) = \{x, P(x), P(P(x)), \ldots s\}. \qquad (1)$$

Элементы (1) упорядочены по невозрастанию значений расстояния от корня дерева путей. Альтернативы формирования дерева кратчайших путей отражаются

листьями, путь от корня до которых не обязательно кратчайший, но восстанавливается по правилу построения $p(x,s)$.

Обозначим L_k – множество листьев текущего дерева на этапе k, L_k^* – подмножество листьев без постоянной пометки. Очевидно, что в любой момент построения дерева кратчайших путей его узлы можно отобразить на элементы множества

$$L_k = \bigcup \{p(s,x), x \in L_k\} \qquad (2)$$

Расширение дерева кратчайших путей происходит только из некоторого листа x_k без пометки, $x_k = \arg\min_x \{d(x), x \in L_k^*\}$. Лист x_k превращается в узел ветвления после операции $L_{k+1}^* = L_k^* \setminus \{x_k\}$ и до окончания поиска становятся пассивным (получает постоянную метку). Ветвление из узла x_k может привести к включению новых или коррекции в сторону уменьшения расстояния от корня s дерева существующих в L_{k+1}^* листьев из множества $\{x_k'\}$. Так как каждому элементу x, $x \in L_k$, соответствует $d(x)$ – длина пути до корня s, то элементы множества $L_k^* = \{x \mid d(x) \geq d(x_k), x \in L_k\}$ представляют приоритетную очередь [1,4].

Очевидно, элементы $x \in L_k \setminus L_k^*$ остаются упорядоченными по невозрастанию значений $d(x)$. Таким образом, необходимо отображение ассоциаций $x \to p(x)$, $x \to d(x)$ и $d(x) \to x$. Для этого представим переменные состояния процесса построения дерева кортежами тернарного отношения $T(x,d,p)$, где x – номер узла дерева, d – расстояние от корня до узла x, p – номер предшествующего узла не обязательно кратчайшего пути в узел x.

Состояние процесса построения дерева на любом его этапе k представлено тройками $(x, p(x), d(x)), x \in L_k^*$. Начальное состояние процесса построения дерева соответствует тройке $(s,s,0)$. Условие завершения процесса – $(L_k^* = \varnothing)$ или $(x_k = t)$, если t – конечная вершина пути. В любом случае $k \leq |N|$ после завершения поиска пути $s \to t$.

Различные виртуальные представления отношения $T(x,d,p)$ отличаются выражениями ключей упорядочения кортежей. В качестве таких ключей могут быть $x+y$, $p+x$ и d.

Виртуальное представление отношения может определяться разными способами, конкретизируемыми в среде программирования баз данных. Например, поддержка понятия структурных индексов в системах семейства FoxPro обеспечивает возможность отображения нескольких законов упорядочения одной таблицы отношения. Кортежи такой таблицы наряду с явно идентифицируемыми полями имеют два часто полезных дополнительных атрибута:

k^0 – номер кортежа в таблице на физическом уровне (выбирается функцией $recno()$);

k^O – номер кортежа в порядке последовательного перечисления кортежей в соответствии с заданным ключом o законом упорядочения (обычно явно не задается, но в процессе перемещения любым способом по кортежам корректность номера текущего кортежа выявляется функциями $bof()$ и $eof()$ для обнаружения выхода за пределы допустимых номеров строк существующей таблицы).

Навигация по кортежам любого представления реализуется операторами поиска по ключу (либо части составного ключа), перехода на начало или конец таблицы, сканирования кортежей по заданным логическим условиям. В любой момент времени

некоторый кортеж таблицы представляет текущую позицию с определенным значением k^0. Изменение значения любого ключа упорядочения меняет соответствие $k^0 \leftarrow k^O$ без изменения k^0 и выражения (2).

Дерево кратчайших путей для приведенного примера графа (рис. 1) в табличной форме задается вариантами представлений (рис. 2), отличающимися законами упорядочения строк:

T_0 – размещение кортежей в памяти на физическом уровне;

T_x – справочник номеров (идентификаторов) начальных вершин дуг графа;

T_d – таблица значений расстояния до вершин графа от его корневой вершины;

T_p – справочник номеров начальных и конечных вершин дуг графа (инверсия графа дерева кратчайших путей, требующаяся для реализации алгоритмов реоптимизации).

```
create table Trace (x C(3), p C(3), d N(3), z N(3))
index on x+y tag x && Справочник вершин дерева
index on d tag d && Приоритетная очередь
index on p+x tag p && Инверсия дерева кратчайших путей
*
do spt with "b"
*
*
select Trace
set order to 0
do show with "T0"
set order to x
do show with "Tx"
set order to d
do show with "Td"
set order to p
do show with "Tp"
*
proc show
 para t
 brow title t;
  fields r0=recno(),x,p,d
```

T0

R	X	P	D
1	b	b	0
2	a	c	6
3	c	b	2
4	e	c	7
5	g	d	12
6	d	a	10
7	f	e	14

TX

R	X	P	D
2	a	c	6
1	b	b	0
3	c	b	2
6	d	a	10
4	e	c	7
7	f	e	14
5	g	d	12

TD

R	X	P	D
1	b	b	0
3	c	b	2
2	a	c	6
4	e	c	7
6	d	a	10
5	g	d	12
7	f	e	14

TP

R	X	P	D
6	d	a	10
1	b	b	0
3	c	b	2
2	a	c	6
4	e	c	7
5	g	d	12
7	f	e	14

Рис. 2. Представления результата построения дерева кратчайших путей

Поддержка понятия структурных индексов позволит далее представить агоритмы мониторинга состояния сети средствами программирования реляционных баз данных.

Алгоритм решения задачи мониторинга изменений

Формирование в лесах подграфов графа транспортной сети дерева кратчайших путей с корнем в вершине s проводится по схеме Дейкстры в представлении T_d:

```
proc spt && Создание дерева кратчайших путей
 para s && Корень дерева
 insert into Trace (x,p,d) values (m.s,m.s,0)
 priv fin
 m.fin=reccount("fin") && Количество целевых вершин
 do dej with m.s && Построение дерева кратчайших путей
retu
```

Сканирование приоритетной очереди и пометка выбираемых вершин преследует цель построения дерева кратчайших путей, определяемого представлением T_x:

```
proc dej && Построение или перестройка дерева кратчайших путей
 para s && Начальный узел ветвления
 select Trace
 set order to x
 if seek(m.s) && Узел ветвления дерева найден в очереди
  priv r
  select Trace
  set order to d && Контекст приоритетной очереди
  scan rest while fin(x) for seek(x,"Graph") nooptimize
   m.r=recno() && Фиксация позиции начала очереди
   set order to x && Контекст листьев дерева
   do spi && Расширение дерева кратчайших путей
   set order to d && Контекст приоритетной очереди
   goto record m.r && Возврат в начало очереди
  ends
 endi
retu
```

Процесс сканирования прерывается после достижения всех целевых вершин, перечисленных во множестве $Y = \{y \mid (?,?,y) \in F$. При этом целевые вершины могут встречаться на кратчайшем пути несколько раз и в любой момент времени исключаться из Y. Контроль необходимости продолжения поиска целевых вершин проводится после постоянной пометки очередной вершины:

```
proc fin && Контроль полноты создаваемого дерева
 para x && Узел ветвления
 if seek(m.x,"fin") && Проверка достижения целевой вершины
  m.fin=m.fin-1
  do notate with m.x,fin.a && Уведомление агента об изменении пути
 endi
retu m.fin>0
```

После выборки очередного узла дерева из очереди его позиция в таблице отношения T_d фиксируется и в контексте отношения T_x проводится расширение дерева из текущей вершины:

```
proc spi && Расширение дерева кратчайших путей
 priv x,d,e
 scatter memvar fields x,d && Выборка листа ветвления
 select Graph
 scan rest while x=m.x && Перебор альтернатив ветвления
  m.e=m.d+w
  select Trace
  if seek(Graph.y) && Вершина уже включена в дерево
   if d>m.e && Обновление пути к существующему листу
    repl p with m.x, d with m.e, z with m.d
   endi
  else && Создание нового листа дерева
   insert into Trace (x,p,d,z) values (Graph.y,m.x,m.e,m.d)
  endi
 ends
 select Trace
retu
```

Изменения дуг сети, определяемые отношением $R(x,y,w)$, должны быть учтены

```
proc ana && Обработчик факта изменения дуг графа сети
 priv d,x
 m.d=infinity
 set order to x in Trace
 select Restr
 scan all && Просмотр измененых дуг графа сети
  select Graph
  if seek(Restr.x+Restr.y) && Измененная дуга принадлежит графа сети
   repl w with Restr.w && Коррекция веса дуги
  else
   insert into Graph (x,y,w) values (Restr.x,Restr.y,Restr.w) && Включение новой дуги
  endi
  select Restr
  if seek(x+y,"Trace").and.(m.d>Trace.d) && Измененная дуга уже в дереве
   m.d=Trace.d && Расстояние до корня изменяемого поддерева
   m.x=Restr.x && Корень изменяемого поддерева
  endi
 ends
 use
 do spr && Организация процесса реоптимизации
retu
```

в новой версии графа $G_x(x,y,w)$. Кроме этого, из дерева, заданного представлением T_x, должно быть удалено поддерево с корнем $x_R = \arg\min_k \{d_k \mid (k \in T_x) \wedge (k,?,?) \in R)\}$:

Далее для краткости изложения приведен пример реализации простейшей версии организации процесса реоптимизации – возврат в состояние выборки вершины x_r из очереди и продолжение процесса ветвления по схеме Дейкстры для измененной части исходного дерева:

```
proc spr && Организация процесса реоптимизации
 select Trace
 set order to 0
 repl all d with infinity for z>m.d && Удаление изменяемых дуг дерева
 count all to m.fin for (d<m.d).and.seek(x,"fin") && Учет оставшихся целевых вершин
 do dej with m.x && Перестройка удаленной части дерева
retu
```

Такая организация характеризуется надежностью учета любых расширений условий ветвления и ограничений на структуру кратчайших путей, а ее вычислительная сложность в среднем в два раза лучше повторного применения алгоритма Дейкстры. Однако последняя процедура может быть заменена одной из известных процедур реоптимизации [2-4] текущего дерева кратчайших путей по представлениям T_x и T_p (в случае выполнения условий их корректного применения).

Итак, регулярная обработка фактов изменения дуг графа сети рассмотренными выше процедурами приводит к уведомлению зарегистрированных агентов об изменении структуры интересных им кратчайших путей.

Заключение

Использование реляционной модели позволяет обеспечить независимость ее представлений от схемы идентификации вершин. Нет ограничений на отражение различных эвристик повышения эффективности реализации идей алгоритма Дейкстры [4].

Представленные результаты позволяют обеспечить слежениt за изменением результатов оптимизации путей на графах транспортных сетей, обеспечивая минимальные задержки на обработку особых событий. Сокращение времени получения оптимального решения в первом приближении пропорционально объему локальных изменений данных задачи. Дополнительная память для хранения индексов таблиц виртуальных представлений не превышает объема $O(m+n)$. При этом сокращается потребность в информационном взаимодействии между потребителями результатов оптимизации путей.

Список литературы / References

16. Steenbrink P.A. Optimization of transport networks. Wiley, 1974. 325 p.
17. Pallottino M.S., Scutelli M.G. A new algorithm for reoptimizing shortest paths when the arc costs change// Operations Research Letters. 2003. Vol. 31, iss. 2. P. 149-160.
18. Hooks E., Yang B., Updating paths in time-varying networks given arc weight changes//Transportation Science. 2005. Vol. 39, iss. 4. P. 451-464.
19. Shortest paths on dynamic graphs: a survey/ Ferone D. [et al]//Pesquisa Operacional. 2017. Vol. 37, iss. 3. P. 487-508.

Адрес для корреспонденции

220013, Республика Беларусь,
г. Минск, ул. П. Бровки, д. 6,
Белорусский государственный университет
информатики и радиоэлектроники
тел. +375-17-293-86-58;
e-mail: rmp@bsuir.by
Ревотюк Михаил Павлович

Address for correspondence

220013, Republic of Belarus,
Minsk, P. Brovka st., 6,
Belarusian state university of informatics and radioelectronics
tel. +375-17-293-86-58;
e-mail: rmp@bsuir.by
Revotjuk Mikhail Pavlovich

Сведения об авторах

Ревотюк М.П., к.т.н., доцент,
доцент кафедры информационных технологий автоматизированных систем Белорусского государственного университета информатики и радиоэлектроники.

Хаджинова Н.В., старший преподаватель кафедры информационных технологий автоматизированных систем Белорусского государственного университета информатики и радиоэлектроники.

Information about the authors

Revotjuk M.P., PhD, associate professor, associate professor of the information technologies in automated systems department of Belarusian state university of informatics and radioelectronics.

Khajynova N.V., senior lecturer of the information technologies in automated systems department of Belarusian state university of informatics and radioelectronics