ВСТРЕЧНЫЙ ПОИСК КРАТЧАЙШИХ ПУТЕЙ НА ДИНАМИЧЕСКИ ОПРЕДЕЛЯЕМЫХ ГРАФАХ

Ревотюк М. П., Хаджинова Н. В., Кузнецова О. В.
Кафедра информационных технологий автоматизированных систем, Белорусский государственный университет информатики и радиоэлектороники
Минск, Республика Беларусь
E-mail: {rmp, kafitas}@bsuir.by

I. 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 до исчерпания возможности его развития из исходной вершины. В этом случае для каждой вершины x требуется представление множества непосредственно достижимых смежных вершин x', $x' = \{y|w(x, y) \geq 0\}$, где $w(x, y)$ – вес дуги $x \to y, x, y \in N$.

Пространство состояний поиска решения алгоритмом Дейкстры традиционно включает: D – массив расстояний от корня дерева, $|D| = |N|$; P – массив номеров предшествующих вершин, $|P| = |N|$; Q – очередь вершин, $|D| \leq |N|$, где элементы упорядочены по текущему значению расстояния от корня дерева [1,2].

На практике возникает потребность поиска путей на графах с изменяемой структурой, когда описание графа задана легко модифицируемым списком дуг. Реляционная модель графа и пространства поиска решения – основа эффективной реализации поиска дерева путей методом бутстрэппинга[2]. Такой метод отличается ограничением потребности в памяти количеством фактически просматриваемых вершин без оценки предельных значений $|N|$, а также возможностью реализации открытых для расширения систем.

Предмет обсуждения – развитие и обобщение приемов реализации метода бутстрэппинга [2] в задачах встречного поиска пути между парами вершин графа с переменной структурой.

II. Модель пространства состояния поиска

Дерево кратчайших путей – связный граф по определению. Если s – исходная вершина, а x – произвольный узел или лист дерева путей, $s, x \in N$, то после завершения поиска последовательность посещаемых вершин обратного пути

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

после тривиального изменения направления перечисления элементов представляет искомый кратчайший путь. Элементы (1) упорядочены по значениям 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 $L(s)$ – множество вершин текущего дерева, L^* – подмножество листьев без постоянной пометки [1]. Очевидно, что в любой момент построения дерева кратчайших путей его узлы можно отобразить на элементы множества

$$L(s) = \bigcup_{x \in N} p(s, x). \qquad (2)$$

Расширение дерева кратчайших путей происходит только из некоторого листа без пометки, а листья из множества $L(s) \setminus L^*$ представляют лишь исторический интерес и становятся пассивными. Так как каждому элементу $x, x \in L(s)$, соответствует $d(x)$ – длина кратчайшего пути до корня s, то парами $(x, d(x))$ активные элементы множества L^* представляют приоритетную очередь [1], а пассивные элементы из $L(s) \setminus L^*$ – постепенно растущее дерево кратчайших путей. Обычно на практике $|L(s)| \ll |N|$, что объясняет привлекательность схемы бутстрэппинга.

Представим переменные состояния процесса построения дерева путей элементами отношения $T(x, d, p)$, где x – номер узла дерева (уникальный элемент), d – расстояние от корня s до узла x, p – номер предшествующего узла кратчайшего пути в узел x. По определению, $L(s)$ – проекция отношения $T(x, d, p)$ по атрибуту x.

Очевидно, что состояние процесса построения дерева на любом этапе k представлено тройкой $(x_k, d(x_k), p(x_k))$. Начальное состояние процесса построения дерева с корнем s соответствует тройке $(s, 0, s)$. Условие завершения процесса – $(L^* = \oslash)$ или $(x_k = t)$, если t – конечная вершина пути. В любом случае $k \leq |L(s)|$ после однонаправленного поиска пути $s \to t$. Нумерация состояний в явном виде не потребуется, если в операции выборки очередного элемента x_k из L^* контролировать условие завершения процесса. Результат операции $L^k \ominus x_k$ – истинность условия $(L^k \neq \oslash)$, новое значение x_k в случае $(L^k \neq \oslash)$ и $k \leftarrow k + 1$. Так как по определению $d(x_k) \leq d(x_{k+1})$, то операция $'\ominus'$ требует упорядочения элементов $L(s)$ по ключу $d(x) \odot x$ (здесь и далее символ $'\odot'$ обозначает операцию конкатенации). Операцию расширения дерева кратчайших путей обозначим символом $' \oplus'$. Результат

операции $L^k \oplus x_k$ – истина, если $x_k \notin L(s)$ и выполнено $L^k \leftarrow L^{k-1} \cup \{x_k\}$. Ее реализация требует упорядочения элементов $L(s)$ по ключу x. Алгоритм Дейкстры поиска кратчайшего пути $s \to t$ в терминах определенных выше операций имеет вид:

```
function sp1(s,t) begin
  T1(x,d,p) = (s,0,s);
  while (L* ⊖ x) ∧ (x ≠ t) do
    foreach (y ∈ x') do
      r = d(x) + w(x,y);
      if (L* ⊕ y) ∨ (d(y) > r) then
        d(y) = r; p(y) = x
      end
    end
  end
end
```

Здесь предполагается обычно делегируемое системам управления базами данных автоматическое переключение между виртуальными представлениями отношения $T(x,d,p)$ по значениям ключей упорядочения их кортежей.

III. Модель и алгоритм встречного поиска

Встречный поиск предлагается проводить на виртуальных преставлениях графа, а отношение $T(x,d,p)$ дополнить атрибутами:

$q(x)$ – признак вершины исходного ($q = 1$) или инвертированного ($q = 2$) графа, $x \in L$;

$z(x)$ – признак постоянной пометки вершины графа, $z(x) = (x \in L \setminus L^*)$.

Ключами виртуальных представлений пространства поиска $T(x,d,p,q,z)$ будут $x \odot q(x)$ и $d(x) \odot x \odot q(x)$. Используя сокращения выражений $x \odot q(x) \equiv x_q$, $(3 - q(x)) \equiv \overline{q}(x)$ и $x \odot \overline{q}(x) \equiv \overline{x}_q$, представим алгоритм встречного поиска кратчайшего пути $s \to t$ в виде:

```
function sp2(s,t) begin
  T1(x,d,p,q,z) = (s,0,s,1,false);
  T2(x,d,p,q,z) = (t,0,t,2,false);
  while L* ⊖ x_q do
    z(x_q) = true;
    if (q(x̄_q) = q̄(x_q)) ∧ z(x̄_q) then
      spt(x_q);
      break;
    end
    foreach (y ∈ x'_q) do
      r = d(x_q) + w(x_q,y);
      if L* ⊕ y then
        d(y) = r; p(y) = x_q;
        q(y) = q(x_q); z(y) = false;
      else
        if (d(y) > r) then
          d(y) = r; p(y) = x_q
        end
      end
    end
  end
end
```

Здесь источник элементов множества x'_q – исходный или инвертированный граф, указывается признаком $q(x_q)$ вершины ветвления x_q. На уровне хранения такие графы заданы списком дуг $G(x,y,w)$, где w – вес дуги $x \to y$. Индексируя отношение G по ключу x или y, получаем представление исходного или инвертированного графа в виде структуры смежности $\{(x,x')\}$ или $\{(y,'y)\}$ (здесь $'y = \{x | w(x,y) \geq 0\}$, где $x, y \in N$).

После достижения некоторой вершины в сопряженном графе процесс расширения встречных деревьев кратчайших путей завершается. Однако для получения оптимального пути необходим перебор всех альтернатив слияния листьев и узлов встречных деревьев с целью поиска вершины v, для которой сумма длин путей $s \to v$ и $v \to t$ имеет минимальное значение f:

```
function spt(x_q) begin
  v = x_q; f = d(v) + d(v̄);
  while L* ⊖ x_q do
    r = d(x_q) + d(x̄_q);
    if f > r then
      v = x_q; f = r;
    end
  end
  do spl(v);
end
```

Из вершины v остается инвертировать путь $v \to t$ для представления пути $t \to s$ в виде (1):

```
function spl(x) begin
  ď = d(x_0); p = p(x_1);
  while p ≠ t do
    d(p) ← ď + d(x_1) − d(p); ď = d(p);
    q(p) = 0; p(p) = x;
    x = p; p = p(x_1);
  end
end
```

Рассмотренный алгоритм поиска построен на основе компактной реляционной модели, допускающей в реальном времени динамическое изменение структуры и параметров графа. Кардинальность отношений определяется фактическим размером исходных данных и результатов поиска. Очевидно, что расширения модели для учета различных ограничений не требуют изменения предложенной схемы поиска решения.

Список литературы

1. Ferone, D. Shortest paths on dynamic graphs: a survey/D. Ferone [et al.]//Pesquisa Operacional, 2017. – Vol. 37, iss. 3. – P. 487–508.

2. Ревотюк, М. П. Встречный поиск кратчайших путей на больших динамических графах методом бутстрэппинга/М.П. Ревотюк, Н.В. Хаджинова //BIG DATA and Advanced Analytics: сб. материалов VI Междунар. науч.-практ. конф., Минск, 20–21 мая 2020 г. В 2 ч. Ч. 1. – Минск: Бестпринт, 2020. – С. 324–331.