Сетевые модели выявления конфликтов координируемых процессов

Ревотюк М.П.; Наймович В.В.
Кафедра информационных технологий автоматизированных систем
Белорусский государственный университет информатики и радиоэлектроники
Минск, Республика Беларусь
e-mail: rmp@bsuir.by

***Аннотация*—Задача выявления конфликтов взаимодействующих процессов на транспортных сетях рассмотрена как задача поиска кратчайших путей на графе сети с ограничениями на структуру пути.**

Ключевые слова: сетевые модели; транспортные операции; координация взаимодействия

Объект рассмотрения – задачи планирования и координации взаимодействующих процессов на транспортных сетях. Пусть транспортная сеть представлена нагруженным ориентированным графом $G(M,N)$, где N и M – множества вершин и дуг графа, а каждой дуге $(i,j)\in M$, $i,j\in N$, соответствует положительное вещественное число $w(i,j)<\infty$, называемое длиной дуги. Если между вершинами s и t, $s,t\in N$, существует ориентированный путь, то длиной пути будем называть сумму длин дуг, образующих путь.

Обозначим для любой дуги $(i,j)\in M$ графа $G(M,N)$ множество допустимых вершин для развития путей из вершины j через $cont(i,j)$, а для любой вершины $x\in N$ множество непосредственно достижимых смежных вершин - x': $x'=\{k|w(x,k)\geq 0\}$.

Очевидно, что $cont(i,j)\subseteq j'$, а в случае отсутствия ограничений на выбор пути после прохождения дуги $(i,j)-cont(i,j)\equiv j'$, $i,j\in N$, $(i,j)\in M$.

Требуется найти, если существует, на множестве вершин $\{s_0=s$, $s_1\in cont(s,s)$, $s_2\in cont(s_0,s_1)$,… $s_i\in cont(s_{i-2},s_{i-1})$,… t} кратчайший путь от вершины s к вершине t, где s, t ∈N.

Содержательный смысл функции $cont(i,j)$ может иметь отношение, например, к правилам дорожного движения, определяющих альтернативы выбора дуг в узле сети с учетом параметров транспортного средства. Другой пример – задачи управления подвижными единицами промышленного транспорта, когда необходимо учесть занятость участков сети на интервалах уже реализуемых маршрутов перевозки. Введение функции $cont(i,j)$ означает намерение разделения модели транспортной сети и модели перемещения по ее дугам.

Ограничения на выбор пути после прохождения дуги $(i,j)\in M$ формально можно рассматривать как изменение структуры графа. Отсюда следует, что искомый алгоритм может быть построен путем целенаправленной модификации алгоритма Дийкстры [1,2]. Общая схема такого алгоритма соответствует жадной волновой схеме построения дерева кратчайших маршрутов от заданной исходной вершины s до всех остальных. Случай поиска кратчайшего пути до заданной конечной вершины f легко учитывается введением проверки необходимости развития волны после прохождения очередной вершины.

Однако при неудачном выборе метода расстановки пометок вершин при реализации алгоритма Дийкстры можно получить неверный результат [2].

Действительно, пусть дерево кратчайших путей характеризуется множеством предшествующих вершин $\{P(x),\ x\in N\}$. Значение расстояния до вершин дерева обозначим через $\{R(x),\ x\in N\}$. Состояние процесса построения дерева при отсутствии ограничений пусть отражается в очереди вершин $\{Q(x), x\in N\}$. Элементы очереди Q={q1,q2,… qi} в этом случае должны быть упорядочены по значениям расстояний так, что $R(q1)\leq R(q2)\leq\ldots R(qi)$.

Известный алгоритм поиска пути $s\to f$ на основе очереди вершин [2]:

```
for x∈N {
 R(x)=∞;  P(x)=x;
}
R(s)=0; // Фиксация корня дерева
Q=s;
do { // Организация ветвления
 x=Q--;
 if (x==f) break;
 r=R(x);
 for y∈x′
  if (R(y)<r+w(x,y)) {
   if (R(y)<∞) Q-=y;
   R(y)=r+w(x,y), P(y)=x;
   Q+=y;
  }
} while (Q≠∅);
```

На первый взгляд, здесь достаточно заменить выражение $y\in x'$ на $y\in cont(P(x),x)$, и задача поиска кратчайших путей с ограничениями решена.

Однако особенность рассматриваемой задачи – зависимость процесса расширения дерева путей от предыстории. Например, граф с ограничениями может включать цикл, когда пропущенные дуги некоторой вершины могут стать продолжением пути из последующих вершин пути. Как итог, решение о включении вершины в дерево кратчайших путей после ее выборки из очереди не является окончательным, хотя итерации поиска будут завершены.

Альтернативы развития путей, отражающие ограничения, требуют перестройки исходного графа. Можно исключить потребность такой перестройки, если отображать процесс построения дерева

кратчайших путей не на очередь вершин, а на очередь дуг {Q(*x*,*y*), (*x*,*y*)∈*M*}. Позиция дуги в очереди дуг пусть, подобно очереди вершин, соответствует потенциалу ее конечной вершины. Просматриваемая дуга, помещенная в очередь, изменит свое состояние лишь один раз в момент выборки из очереди. Это не только упрощают структуру представления очереди – достаточно использования линейного списка, но и позволяют учесть различные условия развития дерева из любой вершины или дуги.

Алгоритм волновой схемы поиска пути $s \to f$ на основе очереди дуг:

```
for (x∈N) {
 R(x)=∞,  P(x)=x;
}
R(s)=0;
for (x,y)∈A {
 level(x,y)=∞, P(x,y)=location(x,y);
}
for (y∈s') {
 level(s,y)=w(s,y), Q+=(s,y);
}
while (Q≠∅) {  // Расширение дерева дуг
 (x,y)=Q--;
 r=level(x,y), R(y)=r,  P(y)=x;
 if (y==f) break;
 for ((z∈y')&&(y∈cont(y,z))
  if (level(y,z)<∞) {
   level(y,z)=r+w(y,z),
   P(y,z)=location(x,y);
   Q+=(y,z);
  }
 }
```

Недостаток алгоритма поиска на основе очереди дуг – дополнительная память порядка O|*M*|. Однако свобода назначения ограничений на единственном экземпляре графа позволяет решать задачи поиска оптимальных путей перемещения разнородных подвижных единиц в реальном времени.

Рассмотрим, например, случай координации процессов перемещения на сети, когда, наряду с необходимостью поиска кратчайших маршрутов, требуется учесть запрет на использование некоторых дуг в процессе движения по уже запланированным маршрутам.

Анализ методов поиска кратчайших путей показывает, что общей схемой поискового алгоритма (простого перебора, метода динамического программирования, ветвей и границ и др.) является волновой характер сканирования пространства решений. Отсюда следует, что не обязательно конструировать дерево путей, а зафиксировать внимание на фронте волны элементарных переходов. Последнее реализуется методами моделирования процесса на расширениях сетей Петри [2].

Статическим представлением процесса решения является структура данных $Z = \langle X, R, P, s, f \rangle$, где X – множество состояний, подлежащих упорядочению построением дерева путей; P – вектор номеров предшествующего состояния; R – вектор потенциалов состояния или оценок по заданному критерию, s и f – исходное и целевое состояния, $s, f \in X$.

Первоначально на этапе конструирования Z положим $R(i) = \infty$, $P(i) = i$, $i \in X$. После завершения поиска оптимальное решение можно найти обратным движением из состояния *f*: $s, \ldots, P(P(f)), P(f), f$.

Волновой процесс на очереди императивно определяемых состояний Q из начального состояния s пусть моделируется рекуррентным алгоритмом

```
template <class Node>  void Search(Node s) {
 for (Q={s}; !Empty(Q); Wakeup(Q));
}
```

Порождение волны из любого состояния x здесь представлено следующей функцией, учитывающей лишь факт достижения этого состояния

```
template <class Node> void Wakeup(Node x) {
  if (x!=f) {
   Q-=x;
   Iterator next(x);
   while (y=next()) Plan(x,y);
  } else while (!Empty(Q)) Destroy(Q);
 }
```

Функция Wakeup сравнивает текущее состояние с целевым и организует либо планирование всех доступных альтернатив развития волны, либо, если цель достигнута, уничтожение всех ранее запланированных альтернатив. Проблемно-зависимым элементом здесь является итератор альтернатив развития волны. Итератор может учитывать предшествующую траекторию перемещения от исходной вершины, используя *Z*.

Функция планирования Plan отражает намерение реализации элементарного автоматного перехода $x \to y$, $x, y \in X$:

```
template <class Node, class Estimation>
 void Plan(Node x, Node y) {
  Estimation r=R(x)+W(x,y);
  if (R(y)>r) {
   if (R(y)<∞) Q-=y;
   R(y)=r, P(y)=x;
   Q+=y; // Утверждение плана перехода x->y
  }
 }
```

Таким образом, процесс волнового поиска привязан лишь к моментам выхода переходов из активного состояния. Другие состояния доступны для наблюдения и анализа на ассоциированной с каждым переходом структурой *Z*.

[1] Ревотюк М.П., Чан З.А. Реляционные модели задач оптимизации управления на интерпретируемых сетях //Проблемы проектирования и производства радиоэлектронных средств: Сб. Материалов III Межд. научно-технической конференции. В 2-х томах. Том 2. – Новополоцк: ПГУ, 2004. – С.139-141.

[2] Ревотюк М.П., Дарадкех Ю.И., Кирейчук В.А. Поиск кратчайших путей на графах полиморфных сетей с ограничениями//Известия Белорусской инженерной академии, № 1(17)/1, 2004. – С. 126-129.