

ԵՒ ՕՒ ԸՆ ԱԾԵԸԱ

УДК 002.6 + 519.688

Մ. Մ. Մանուկյան

Կարճատև ճանապարհների գտնելու խնդիրը համարյա ինքնավար խնդիրների դասում
իներվալային ճանապարհների գտնելու խնդիրը

(Представлено академиком Ю.Г. Шукурьяном 5/III 2007)

Կարճատև ճանապարհների գտնելու խնդիրը: *ограниченное вершинное покрытие, двудольный граф, IDA*, эвристический поиск*

Մտնելու խնդիրը. Կարճատև ճանապարհների գտնելու խնդիրը համարյա ինքնավար խնդիրների դասում
գրաֆիկայի խնդիրը առաջ է ելնում ինքնավար խնդիրների դասում [1].
Բնութագրենք հետևյալ տարբերակը խնդիրի ձևակերպման.

Ընդհանուր դեպքում $G(V_1, V_2, E)$ և երկու դրական թվեր k_1, k_2 . Գտնել
այնպիսի ենթաբազմություններ $C_1 \subseteq V_1$ և $C_2 \subseteq V_2$, $|C_1| \leq k_1$ և $|C_2| \leq k_2$, որոնց համար
որևէ թիվ $e \in E$ համար e -ի մեկ կամ ևս երկու ծայրերը պատկանում են $C_1 \cup C_2$ և գումար
 $|C_1| + |C_2|$ նվազագույն է.

Այս խնդիրը պատկանում է NP -մեծ խնդիրների դասին [1]. Առաջարկվում է
էվրիստիկ ալգորիթմ, որը հիմնված է IDA^* [2] մեթոդի վրա և ցուցաբերում է
արդյունքներ փորձերի.

Կարճատև ճանապարհների գտնելու խնդիրը IDA^* . Այս ալգորիթմը հիմնված է
գնահատման մեթոդի վրա IDA^* . Գնահատումը կատարվում է
դեռնի, որոշակի գագաթները լուծում են խնդիրը. Երկու
գագաթի n դեռնի ցանցի գնահատումը $f(n) = g(n) + h(n)$, որտեղ
 $g(n)$ - ճանապարհի արժեքը սկզբնական գագաթից n գագաթին, $h(n)$
- էվրիստիկ ֆունկցիա, որը կարող է նվազագույն հեռավորությունը
կազմակերպել լուծման. $f(n)$ ֆունկցիան ցուցաբերում է
ճանապարհի երկարության գնահատումը n գագաթից, որը
անցնում է n գագաթով.

Գնահատումը կատարվում է փոփոխական կերպով, որտեղ
գնահատումը սահմանափակվում է $f(n)$ ճանապարհի
գնահատման խորությամբ. Եթե լուծում է

найденно, то поиск прекращается, в противном случае ограничение $f(n)$ увеличивается на единицу и поиск возобновляется. Преимуществом данного алгоритма является малое использование памяти. Достаточно иметь объем памяти для хранения вершин пути от решения к начальной вершине.

Эвристическая функция $h(n)$ называется допустимой, если она никогда не превышает реальную стоимость до решения. В этом случае поиск является полным (если решение существует, оно найдется) и оптимальным (найденное решение всегда оптимально).

Υάδενήδè÷âñêàÿ ôóí êöèÿ. Вершина дерева поиска характеризуется подмножествами $C_1(n), C_2(n)$ множеств V_1 и V_2 , которые соответствуют промежуточным вершинным покрытиям. Компонента $g(n)$ оценочной функции $f(n)$ в таком случае равна $|C_1| + |C_2|$. Компонента $h(n)$ вычисляется следующим образом. Для всех значений x от 0 до $k_1 - |C_1|$ выбираются x непокрытых вершин (т.е. не принадлежащих покрытию C_1) из множества V_1 с наибольшим количеством непокрытых инцидентных ребер. Количество y вершин из V_2 , необходимое для покрытия, вычисляется таким образом, чтобы гарантировать допустимость эвристической функции.

В ходе алгоритма используются два массива:

leftCover : отсортированный по убыванию массив, элементом которого является количество непокрытых ребер, инцидентных данной вершине из множества V_1 ;

rightCover : отсортированный по убыванию массив, элементом которого является количество непокрытых ребер, инцидентных данной вершине из множества V_2 .

Псевдокод алгоритма вычисления $h(n)$ представлен ниже.

Algorithm 1. Псевдокод вычисления эвристической функции

```

 $h \leftarrow \infty$ 
for  $x \leftarrow 0, k_1 - |C_1|$  do
   $y \leftarrow$  количество ненулевых элементов rightCover
  if  $h > x + y$  then
     $h \leftarrow x + y$ 
  end if
   $k \leftarrow leftCover[x + 1]$ 
  while  $k > 0$  do
     $right \leftarrow$  индекс самого правого ненулевого элемента rightCover
    for  $j \leftarrow right, 1$  do
      if  $k = 0$  then
        break

```

```

    end if
     $k \leftarrow k - 1$ 
     $rightCover[j] \leftarrow rightCover[j] - 1$ 
  end for
end while
end for
return  $h$ 

```

Lemma 2. В случае двудольного графа с максимальной степенью 2 существует полиномиальный алгоритм сложности $O((k_1 + k_2)^3)$ [3]. Легко увидеть, что компонентами связности такого графа являются циклы и простые цепи.

Рассмотрим случай циклов. Если для циклической компоненты C_m длины m справедливо $m \leq 2k_1$ или $m \leq 2k_2$, то компонента может быть покрыта оптимальным образом. Если $m - 2 \leq 2(k_1 + k_2)$, то покрытие может быть построено, используя на одну вершину больше, чем в предыдущем случае. Если $m - 2 > 2(k_1 + k_2)$, то покрытие невозможно.

В случае простых цепей рассматриваются три варианта: (1) цепи нечетной длины, (2) цепи четной длины, начинающиеся и заканчивающиеся в множестве вершин V_1 , (3) цепи четной длины, начинающиеся и заканчивающиеся в множестве вершин V_2 . Цепи четной длины могут быть покрыты оптимально (с точки зрения неограниченного вершинного покрытия), помещая в покрытие каждую вторую вершину начиная со второй. Таким образом, для цепи длины m требуется $m/2$ вершин. Если это невозможно сделать из-за ограничений, то в оптимальное покрытие помещается на одну вершину больше. В случае нечетной длины цепи требуется $\lceil m/2 \rceil$ вершин, неважно в какой пропорции из двух множеств вершин. Поэтому этот случай можно рассматривать в последнюю очередь.

Покрытие в общем случае производится следующим образом [3]. Для каждой компоненты возможно максимум $1 + \min(k_1, k_2)$ покрытий, а число компонент не может быть больше $k_1 + k_2$. На каждом шаге два набора оптимальных покрытий объединяются. Для этого требуется $O((k_1 + k_2)^2)$ вычислений. Размер объединения двух наборов также ограничен $O(k_1 + k_2)$. Таким образом, у нас $k_1 + k_2$ объединяющих шагов, каждый требующий $O((k_1 + k_2)^2)$ вычислений. В итоге получается, что ограниченное покрытие двудольного графа с максимальной степенью 2 может быть выполнено за $O((k_1 + k_2)^3)$ шагов.

Theorem 1. Поиск осуществляется начиная с пустого покрытия. Организация ветвления производится следующим образом.

Выбирается вершина с наибольшей степенью. Если наибольшая степень равна двум, то минимальное покрытие, если оно существует, находится при помощи полиномиального алгоритма, тем самым поиск в этом направлении завершен. Для покрытия ребер, инцидентных выбранной вершине, нужно добавить в покрытие данную вершину либо все инцидентные вершины. Тем самым задача разбивается на две подзадачи. Так как максимальная степень вершины больше или равна 3-м, то задача с размером $k = k_1 + k_2$ разбивается на две подзадачи с размерами $k - 1$ и максимум $k - 3$.

Î öáí èà ðàçì áðà äáðááàà ì îèñèà. Представленный алгоритм работает рекурсивно. Число рекурсий равно количеству вершин в соответствующем дереве поиска. Это количество характеризуется однородным линейным рекуррентным уравнением с константными коэффициентами. Если алгоритм решает задачу размера k и затем рекурсивно вызывает себя для задач с размерами $k - d_1, \dots, k - d_i$, то (d_1, \dots, d_i) называется *вектором разветвления* данной рекурсии. Это соответствует рекурсии

$$t_k = t_{k-d_1} + \dots + t_{k-d_i}. \quad (1)$$

Характеризующим полиномом этой рекурсии является

$$z^d = z^{d-d_1} + \dots + z^{d-d_i}, \quad (2)$$

где $d = \max\{d_1, \dots, d_i\}$. Если α является корнем (2) с максимальным абсолютным значением, то t_k равно α^k с точностью до полиномиального фактора. $|\alpha|$ называется *числом разветвления*, соответствующим вектору разветвления (d_1, \dots, d_i) . Если α является единичным корнем, то $t_k = O(\alpha^k)$. Заметим, что уравнением (1) оценивается количество листьев дерева поиска, которые составляют больше половины всего количества вершин дерева поиска. В нашем случае вектором разветвления является $(1, 3)$, а соответствующим числом разветвления 1.4656 — значение единичного корня уравнения (2). Поэтому размер дерева поиска равен $O(1.4656^k)$, где $k = k_1 + k_2$.

В таблице представлены количества пройденных вершин дерева поиска в версиях программы с использованием эвристики и без нее.

	$ V_1 $	$ V_2 $	k_1	k_2	$ E $	без эвристики	с эвристикой
пример 1	30	30	20	20	60	1525	90
пример 2	40	40	20	20	100	22138	1337
пример 3	40	40	20	20	150	67012	1777
пример 4	50	50	25	25	180	1790949	18074
пример 5	50	50	25	25	150	2216536	32584

Многочисленные эксперименты показывают эффективность использования эвристической функции. Сравнительный анализ, проведенный

на примерах с версией программы без эвристического поиска, наглядно демонстрирует важность выбора надлежащей эвристической функции для практического использования программы.

Институт проблем информатики и автоматизации НАН РА

Գ.Ն. Գեոլեցյան

Երկկողմանի գրաֆում սահմանափակումներով գազաթային ծածկույթ գտնելու Էվրիստիկ ալգորիթմ

Աշխատանքում ներկայացվում է երկկողմանի գրաֆում սահմանափակումներով գազաթային ծածկույթ գտնելու էվրիստիկ ալգորիթմ՝ հիմնված IDA* փնտրման մեթոդի վրա: Տրվում է փնտրման ծառի չափի գնահատականը, և ներկայացվում են փորձնական արդյունքներ:

G.H. Geoletsyan

Heuristic Algorithm for Constraint Bipartite Vertex Cover Problem

An algorithm for constraint bipartite vertex cover based on IDA* heuristic search method is presented. A bound for search tree size is obtained and some experimental results are presented.

Էնթոսթասթոսթոս

1. *Sy-Yen Kuo, W. Kent Fuchs.* - In: DAC. 1986. P. 385-390.
2. *Korf R.E.* - In: IJCAI. 1985. P. 1034-1036.
3. *Fernau H., Niedermeier R.* - J. Algorithms. 2001. N 38 T. 2. P. 374-410.