

УДК 513.25/26

О МАРШРУТИЗАЦИИ СИСТЕМЫ ПОТОКОВ НА ГРАФЕ

М.А. Яковлева

В ряде случаев при исследовании решения транспортной задачи возникает потребность иметь результат не в виде потоков на отдельных участках сети, а в виде перевозок по маршрутам, связывающим пункты производства с пунктами потребления. Вопрос о переходе от одной формы задания потока на сети к другой в теоретическом плане исследован в [1]. Целью настоящей работы является описание на языке АЛГОЛ-60 рабочего алгоритма, позволяющего находить указанные маршруты.

Более точно задача может быть сформулирована следующим образом. Имеется конечный ориентированный граф без направленных циклов. На множестве дуг $\{A_1, A_2, \dots, A_m\}$ этого графа задана вещественная неотрицательная функция $c(A_k)$. Требуется найти некоторое число маршрутов M_1, M_2, \dots, M_z и потоки по ним $p(M_\ell)$, $\ell = 1, 2, \dots, z$, при которых выполнялись бы равенства

$$\sum_{\ell=1}^z a_{k\ell} p(M_\ell) = c(A_k), \quad k=1, 2, \dots, m,$$

где

$$a_{k\ell} = \begin{cases} 1, & A_k \in M_\ell, \\ 0, & A_k \notin M_\ell. \end{cases}$$

При этом маршруты M_ℓ таковы, что каждый из них связывает вершину графа, являющуюся источником, с вершиной, являющейся стоком. Условимся называть источником такую вершину графа, в ко-

торой сумма значений функции c на входящих дугах меньше, чем сумма значений на выходящих дугах, а сток - вершину, в которой сумма значений функции c на входящих дугах больше суммы значений на выходящих дугах.

Такое требование на выбираемые маршруты, как нетрудно видеть, обеспечивает минимизацию $\sum_{\ell=1}^m \rho(M_\ell)$. В собственно транспортной интерпретации это соответствует минимальному объему погрузки.

В общих чертах алгоритм заключается в следующем.

1. Пусть уже определены маршруты $M_1, M_2, \dots, M_{\ell-1}$ с потоками $\rho(M_1), \rho(M_2), \dots, \rho(M_{\ell-1})$. Положим

$$c^{(\ell)}(A_k) = c(A_k) - \sum_{s=1}^{\ell-1} a_{k,s} \rho(M_s).$$

2. Исключим из рассмотрения все дуги A_k , для которых $c^{(\ell)}(A_k) = 0$, и найдем маршрут M_ℓ , связывающий вершину, в которую не входит ни одна из оставшихся дуг, с вершиной, из которой не выходит ни одна дуга.

3. В качестве потока по маршруту M_ℓ возьмем величину

$$\rho(M_\ell) = \min_{A_k \in M_\ell} \{c^{(\ell)}(A_k)\}$$

и вернемся к пункту 1. Так будем продолжать до тех пор, пока для некоторого k не окажется $c^{(\ell)}(A_k) = 0$, $k=1, 2, \dots, m$.

Дадим теперь некоторые пояснения к программе, реализующей этот план. Целые M и N задают, соответственно, число дуг и число вершин рассматриваемого графа, которые мы будем считать перенумерованными. Для задания множества дуг графа предназначены два массива $X[1:M]$ и $Y[1:M]$, значения которых принадлежат множеству $\{1, 2, \dots, N\}$. Дуга графа A_k определяется парой $(X[k], Y[k])$, причем мы будем считать, что дуга направлена из вершины с номером $X[k]$ в вершину $Y[k]$. Для значений функций $c^{(\ell)}$ отведен массив $C[1:M]$, k -ая компонента которого содержит значение функции на дуге $(X[k], Y[k])$. Первоначально в этом массиве хранятся значения функции c . Скалярные переменные $A, A1, K, P$ и $A3$ служат рабочими ячейками.

Программа состоит из двух частей, оформленных в виде процедур ПОРЯДОК и МАРШРУТ. Первая часть является вспомогательной -

в ней выполняется такая перестановка исходных данных, которая позволяет экономно организовать работу второй части программы.

Мы будем называть порядок расположения дуг хорошим, если для каждой дуги все дуги, входящие в вершину, из которой эта дуга выходит, лежат выше нее. В процедуре ПОРЯДОК выполняется перестановка дуг в массивах, после которой порядок их расположения становится хорошим. Сначала для каждой вершины подсчитывается число дуг, входящих в эту вершину. Полученные числа запоминаются в рабочем массиве С4. Часть процедуры, следующая за меткой Ю, работает многократно, причем при каждом повторении считается, что дуги, стоящие на местах с I-го по (N - I)-ое, уже удовлетворяют требованию на порядок и что в массиве С4 для каждой вершины содержится число входящих в эту вершину дуг, которые не поставлены еще на свое окончательное место, то есть дуг, стоящих на местах с N-го по M-ое. Среди неупорядоченных дуг отыскивается дуга, для которой $C4[X\{K\}] = 0$, то есть дуга, выходящая из вершины, в которую не входит ни одна из неупорядоченных дуг. Эта дуга помещается на место N, и при этом порядок расположения дуг, стоящих на первых N местах, оказывается хорошим. Для вершины, в которую входила перенесенная на место N дуга, значение С4 уменьшается на единицу. Работа процедуры заканчивается, когда все дуги оказываются упорядоченными.

В процедуре МАРШРУТ последовательно определяются маршруты и потоки по ним. Внешний цикл этой процедуры ведает выбором начал маршрутов. Как уже указывалось при общем описании алгоритма, мы принимаем во внимание лишь дуги, для которых $c^{(0)}(A_k) > 0$. Первая такая дуга, встретившаяся при последовательном просмотре списка дуг, выбирается в качестве начальной дуги первого маршрута. После определения маршрута и выполнения всех связанных с этим операций происходит возврат на метку Ю, то есть проверяется, не можем ли мы же самая дуга, которая была выбрана за начальную для предыдущего маршрута, послужить начальной и для следующего маршрута. Лишь в том случае, если это не так, происходит переход к исследованию очередной дуги. Очевидно, что благодаря проведенному упорядочению, начальным вершины тех выбранных маршрутов, удовлетворяют требованию, сформулированному в пункте 2.

Логический массив F[I:M], локализованный в этой процедуре, является рабочим массивом. Перед отысканием очередного маршрута

Me всем элементам этого массива придается значение НОЛЬ. В дальнейшем в массиве *P* отмечаются дуги, вошедшие в маршрут, что дает возможность упростить цикл, выполняющий вычисление очередной очередной функции $c^{(i)}$.

Легко видеть, что при имеющемся порядке расположения дуг каждая последующая дуга любого маршрута лежит в списке дуг либо предыдущей. Это дает возможность определить все дуги маршрута за один просмотр списка дуг, начиная с дуги, выбранной для данного маршрута за начальную, и кончая последней дугой списка. Этот основной цикл расположен в теле процедуры следом за выводом на печать начальной вершины маршрута. Все вершины определяемого маршрута печатаются сразу же по мере их нахождения. В этом же цикле попутно определяется поток по маршруту, который помещается в рабочую ячейку *AB* и выводится на печать следом за последовательностью вершин маршрута. После потока печатается число -333, которое отделяет один маршрут от другого.

Ниже приведена программа описанного алгоритма.

начало целый *M, N, A, AI, K, P*;

реальный *AB*; ввод (*M, N*);

начало целый массив *X[1:M], Y[1:M]*;

реальный массив *C[1:M]*; ввод (*X, Y, C, AB*);

начало

процедура ПОРЯДОК;

начало целый массив *СЧ[1:N]*; для *K:=1* шаг *I* до *N* цикл *СЧ[K]:=0*;

для *K:=1* шаг *I* до *M* цикл *СЧ[Y[K]]:=СЧ[Y[K]]+1*; *P:=I*;

0; для *K:=P* шаг *I* до *M* цикл если *СЧ[X[K]] = 0* то

начало *СЧ[Y[K]]:=СЧ[Y[K]]-1*; *A:=X[K]*; *X[K]:=X[P]*;

X[P]:=A; *A:=Y[K]*; *Y[K]:=Y[P]*; *Y[P]:=A*;

AB:=C[K]; *C[K]:=C[P]*; *C[P]:=AB*; *P:=P+1* конец;

если *P < M* то на *P* монти;

процедура МАРШРУТ;

начало логический массив *P[1:M]*; для *P:=1* шаг *I* до *M* цикл

0; если *C[P] > 0* то начало для *K:=1* шаг *I* до *M* цикл

P[K]:=НОЛЬ; *AB:=C[P]*; *A:=X[P]*; вывод (*A*);

для *K:=P* шаг *I* до *M* цикл если *X[K] = A* & *C[K] > 0* то

начало *A:=Y[K]*; вывод (*A*);

если *C[K] < AB* то *AB:=C[K]*; *P[K]:=ИСТИНА* конец;

A:= -333; вывод (*AB, A*); для *K:=P* шаг *I* до *M* цикл

если P[K] то C[K]:= C[K] - AB; на D конец конец;
ПОРЯДОК; МАРШРУТ конец конец конец."

Л и т е р а т у р а

1. Л.Р.Ферд, Д.Р.Фалкерсон, Цепочки в сетях. "Мир", М., 1966.

Поступила в редакцию

1.11. 1971 г.