УДК 519.876.3

Прокудин Г. С., канд. техн. наук

МЕТОДЫ ОПТИМИЗАЦИИ ПЕРЕВОЗОК В ТРАНСПОРТНЫХ СЕТЯХ

Национальный транспортный университет

В статье описаны новые методы решения открытых сетевых транспортных задач, а именно: метод нахождения кратчайших маршрутов на транспортной сети и метод пропорционального перераспределения объемов перевозок между участниками перевозочного процесса. Анализ известных аналогичных методов выявил ряд существенных недостатков, которые в значительной степени сужают область их использования. Предложенные методы лишены выявленных недостатков, прошли экспериментальную проверку и показали при этом свою надежность и эффективность по сравнению с существующими подходами.

Введение

Одной из первоочередных задач организации экономически выгодных перевозок грузов различными видами транспорта является задача определения наиболее рациональных маршрутов перевозок в заданной топологии сети поставщиков и получателей грузов. С точки зрения теории графов подобные задачи классифицируются как задачи поиска на ориентированных либо неориентированных (в зависимости от решаемой задачи) графах оптимальных связей на заданной матрице корреспонденций.

Не менее актуальной является также задача оптимизации транспортных перевозок грузов в условиях не соответствия объёмов его предложения объёмам его спроса. В этом случае принято говорить об открытых или не сбалансированных перевозках.

В связи с тем, что в большинстве случаев грузовые перевозки производятся на густо разветвлённой сети коммуникаций и в условиях их не сбалансированности, то совместное решение этих задач позволит получить хороший экономический эффект.

Анализ публикаций

Наиболее распространенной на практике является задача поиска оптимального маршрута на сети по критерию кратчайшего расстояния [1]. Эта задача естественным образом моделируется с помощью связной сети (графа) G, в которой каждому ребру (дуге) приписан положительный вес, равный длине ребра. Длина пути в такой сети равна сумме длин ребер, составляющих этот путь. В терминах сетей задача сводится к отысканию кратчайшего пути между любыми двумя заданными вершинами графа G.

Задачи о кратчайших путях относятся к фундаментальным задачам комбинаторной оптимизации [2], т.к. многие такие задачи можно свести к отысканию кратчайшего пути на сети. Существуют различные типы задач о кратчайшем пути:

- между двумя заданными вершинами;
- между данной вершиной и всеми остальными;
- между каждой парой вершин в сети;
- между двумя заданными вершинами для путей, проходящих через одну или несколько указанных вершин;
- первый, второй, третий и т.д. кратчайшие пути в сети.

Наибольший интерес для решения вышеназванных сетевых транспортных задач из перечисленных типов представляют первые три, причем первые два из них реализуются с помощью разновидностей алгоритма Дейкстры, а третий с помощью алгоритма Флойда.

Данная статья посвящена анализу указанных алгоритмов и разработке нового, который должен быть более экономичным (в части объема необходимых вычислительных процедур по сравнению с существующими) и способным решать сетевые задачи большой размерности.

Задача нахождения кратчайшего пути между двумя заданными вершинами

Пусть есть ориентированный граф $G = (V, E)$ (в дальнейшем сокращенно орграф), у которого все дуги имеют неотрицательные метки (стоимости дуг), а одна вершина определена как *источник*. Задача состоит в нахождении стоимости кратчайших путей от источника ко всем остальным вершинам графа G. Эта задача часто называется *задачей нахождения кратчайшего пути с одним источником*. Также отметим, что мы будем говорить о длине пути даже тогда, когда она измеряется в других, не линейных, единицах измерения, например во временных единицах.

Можно представить граф G в виде карты маршрутов рейсовых полетов самолетов из одного аэропорта (а/п) в другой, где каждая вершина соответствует а/п, а дуга $v \longrightarrow w$ - рейсовому маршруту из а/п v в а/п w. (Метка дуги $v \longrightarrow w$ - это время полета из а/п v в а/п w). При этом может возникнуть предположение, что в данном случае в качестве модели больше подходит неориентированный граф, поскольку метки дуг $v \longrightarrow w$ и $w \longrightarrow v$ могут совпадать. Но фактически в большинстве случаев время полета в противоположных направлениях между двумя а/п различно. Кроме того, допущение о совпадении меток дуг $v \longrightarrow w$ и $w \longrightarrow v$ принципиально не влияет на решение поставленной задачи.

В этом случае решение *задачи нахождения кратчайшего пути с одним источником* формулируется как минимальное время перелетов между различными а/п. Для решения поставленной задачи будем использовать "жадный" алгоритм, который часто называют *алгоритмом Дейкстры* (*Dijkstra*) [3].

Алгоритм строит множество S вершин, для которых кратчайшие пути от источника уже известны. На каждом шаге к множеству S добавляется та из оставшихся вершин, расстояние до которой от источника меньше, чем для других оставшихся вершин. Если стоимости всех дуг неотрицательны, то можно быть уверенным, что кратчайший путь от источника к конкретной вершине проходит только через вершины множества S. Назовем такой путь *особым*. На каждом шаге алгоритма используется также массив D, в который записываются длины кратчайших особых путей для каждой вершины. Когда множество S будет содержать все вершины орграфа G, т.е. для всех вершин будут найдены "особые" пути, тогда массив D будет содержать длины кратчайших путей от источника к каждой вершине.

Алгоритм Дейкстры представлен в виде псевдокода на листинге 1, где предполагается, что в орграфе G вершины поименованы целыми числами, т.е. множество вершин $V = \{1, 2, ..., n\}$, причем вершина 1 является источником. Массив C - это двумерный массив стоимостей, где элемент $C[i, j]$ равен весу дуги $i \longrightarrow j$. Если дуги $i \longrightarrow j$ не существует, то $C[i,j]$ полагается равной ∞ (бесконечности). На каждом шаге $D[i]$ содержит длину текущего кратчайшего особого пути от вершины 1 к вершине i.

Пример 1. Используем алгоритм Дейкстры для орграфа, показанного на рис. 1.(Для этого примера массив C будет иметь вид, представленный в таблице 1).

Листинг 1. Алгоритм Дейкстры

```
procedure Dijcstra;
begin
(1) S := {1};
(2) for i:= 2 to n do
(3)   D[i]:=C[l,i]; {инициализация D}
(4) for i:= 1 to (n-1) do
      begin
```

(5) (выбор из множества V\S такой вершины *w*, что значение D[w] минимально);
(6) добавить *w* к множеству S;
(7) **for** (каждая вершина *v* из множества V\S) **do**
(8) D[v]:= min(D[v],D([w] + C[w,v])
end
end;{ *Dijcstra* }

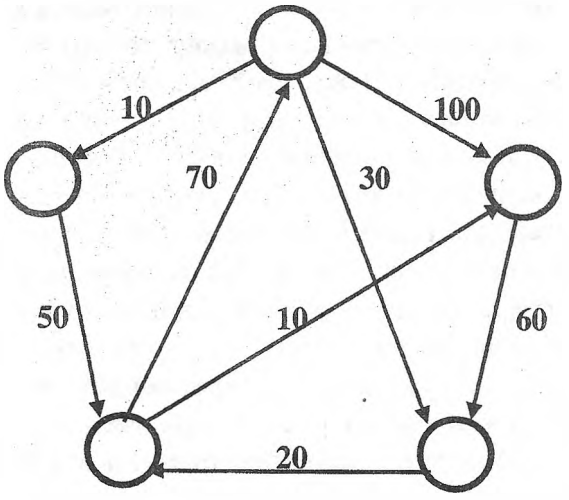

Рис. 1. Орграф с помеченными дугами

В начале S = {1}, D[2] = 10, D[3] = ∞, D[4] = 30 и D[5] = 100. На первом шаге цикла (строки (4) - (8) листинга 1) *w* = 2, т.е. вершина 2 имеет минимальное значение в массиве *D*.

Таблица 1. Массив *C*

C =

	1	2	3	4	5
1	∞	10	∞	30	100
2	∞	∞	50	∞	∞
3	70	∞	∞	∞	10
4	∞	∞	20	∞	∞
5	∞	∞	∞	60	∞

Затем вычисляем D[3] = min(∞, 10 + 50) = 60; D[4] = min(30, 10 + ∞) = 30 и D[5] = min(100, 10 + ∞) = 100. Последовательность значений элементов массива *D* после каждой итерации цикла показана в табл. 2.

Таблица 2. Вычисления по алгоритму Дейкстры на орграфе из рис. 1

Итерация	*S*	*w*	D[2]	D[3]	D[4]	D[5]
Начало	{1}	-	**10**	∞	30	100
1	{1,2}	2	**10**	60	**30**	100
2	{1,2,4}	4	**10**	**50**	**30**	100
3	{1,2,4,3}	3	**10**	**50**	**30**	**60**
4	{1,2,4,3,5}	5	**10**	**50**	**30**	**60**

Примечание: Подчёркнутыми в таблице показаны кратчайшие расстояния от источника (1-ой вершины) до каждой из оставшихся вершин графа.

Задача нахождения кратчайших путей между каждой парой вершин в сети

В этом случае мы сталкиваемся с *общей задачей нахождения кратчайших путей,* т.е. нахождением кратчайших путей между всеми парами вершин орграфа. Более строгая формулировка этой задачи следующая: есть граф *G = (V, E),* каждой дуге *v—>w* этого графа сопоставлена неотрицательная стоимость *C[v,w].* Необходимо найти для каждой пары вершин (*v, w*) путь от вершины *v* до вершины *w,* длина которого минимальна среди всех возможных путей от *v* к *w.*

Существует прямой способ решения данной задачи, использующий *алгоритм Флойда* (*R. W. Floyd*) [4]. Для определенности положим, что вершины графа последовательно пронумерованы от 1 до *n.* Алгоритм Флойда использует матрицу *A* размера *n* x *n,* в которой вычисляются длины кратчайших путей. Вначале *A*[*i, j*] = *C*[*i, j*] для всех $i \neq j$. Если дуга *i—>j* отсутствует, то *C*[*i j*] = ∞ (бесконечности). Каждый диагональный элемент матрицы *A* равен 0.

Над матрицей *A* выполняется *n* итераций. После *k*-й итерации *A*[*i, j*] содержит значение наименьшей длины путей из вершины *i* в вершину *j,* которые не проходят через вершины с номером,

большим k. Другими словами, между концевыми вершинами пути i и j могут находиться только вершины, номера которых меньше или равны k.

На k-й итерации для вычисления матрицы A применяется следующая формула:

$$A_k[i,j] = \min(A_{k-1}[i,j], A_{k-1}[i,k] + A_{k-1}[k,j]).$$

Нижний индекс k обозначает значение матрицы A после k-й итерации, но это не означает, что существует n различных матриц, этот индекс используется для сокращения записи. Графическая интерпретация этой формулы показана на рис.2.

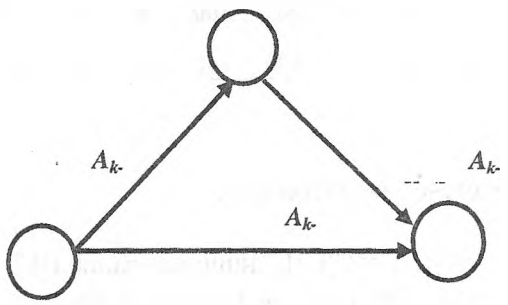

Рис. 2. Включение вершины k в путь от вершины i к вершине j

Пример 2. На рис. 3 показаны значения матрицы A после пяти итераций для графа, изображенного на рис. 1. (Здесь затемненными показаны более короткие маршруты между любыми парами вершин графа, которые появляются на каждой новой итерации). Ниже на листинге 2 представлен псевдокод алгоритма Флойда.

Листинг 2. Алгоритм Флойда

```
procedure Floyd (var A:array[1..n, 1..n] of
    real;C: array[1 .. n,  1 .. n] of real);
var i, j, k:integer;
begin
  for i := 1 to n do
      for j := 1 to n do
          A[i, j] := C[i, j];
  for i := 1 to n do
          A[i, i] := 0;
  for k := 1 to n do
  for i  := 1 to n do
  for j  := 1 to n do
          if A[i, k] + A[k, j] < A[i, j] then
            A[i, j] := A[i, k] + A[k, j];
end; { Floyd }
```

Задача нахождения кратчайших путей между заданными множествами вершин в сети

В качестве исходных данных для этой задачи берем уже известный нам помеченный орграф $G(V,E)$, изображенный на рис. 1, который содержит время полета по маршрутам, связывающим определенные а/п, причем из множества его вершин V *отбираем* подмножество а/п вылетов (m а/п). Требуется отыскать кратчайшие маршруты перелетов между всеми а\п вылетов и посадок, включая и посадки в промежуточных а\п (ими могут быть как а\п вылетов, так и а\п посадок самолетов) [5]. Иными словами мы должны получить матрицу кратчайших расстояний между а\п вылетов и а\п посадок, аналогичную матрице, изображенной в таблице 3.

Таблица 3. Матрица кратчайших расстояний

		А/п посадок			
А/п вылетов		1	2	...	n
	1	C_{11}	C_{12}	...	C_{1n}
	2	C_{21}	C_{22}	...	C_{2n}
	...	...	...	...	...
	m	C_{m1}	C_{m2}	...	C_{mn}

	1	2	3	4	5
1	0	10	∞	30	100
2	∞	0	50	∞	∞
3	70	∞	0	∞	10
4	∞	∞	20	0	∞
5	∞	∞	∞	60	0

$A_0[i, j]$

	1	2	3	4	5
1	0	10	∞	30	100
2	∞	0	50	∞	∞
3	70	80	0	100	10
4	∞	∞	20	0	∞
5	∞	∞	∞	60	0

$A_1[i, j]$

	1	2	3	4	5
1	0	10	60	30	100
2	∞	0	50	∞	∞
3	70	80	0	100	10
4	∞	∞	20	0	∞
5	∞	∞	∞	60	0

$A_2[i, j]$

	1	2	3	4	5
1	0	10	60	30	70
2	120	0	50	150	60
3	70	80	0	100	10
4	90	100	20	0	30
5	∞	∞	∞	60	0

$A_3[I, j]$

	1	2	3	4	5
1	0	10	50	30	60
2	120	0	50	150	60
3	70	80	0	100	10
4	90	100	20	0	30
5	150	160	80	60	0

$A_4[i, j]$

	1	2	3	4	5
1	0	10	50	30	60
2	120	0	50	120	60
3	70	80	0	70	10
4	90	100	20	0	30
5	150	160	80	60	0

$A_5[i, j]$

Рис. 3. Последовательные значения матрицы *A*

Для решения этой задачи рассмотренные алгоритмы не применимы. Алгоритм Дейкстры недостаточен для ее решения (он находит всего лишь одну строку из матрицы кратчайших расстояний), а алгоритм Флойда избыточен, т.к. он генерирует полную матрицу кратчайших расстояний между любыми парами а\п размерностью *n* на *n*. Поэтому рассмотрим принципиально новый алгоритм (*New*), псевдокод которого изображен на листинге 3.

Листинг 3. Алгоритм построения матрицы кратчайших расстояний

```
procedure New(var D:array[1..m, 1..(m+n)] of real; C:array[1..(m+n), 1..(m+n)] of real);
    begin
    (1) for i:=1 to m do
          begin
            S := {i}; { выбор очередной вершины
                        из подмножества а\п вылетов }
            for j:=1 to (m+n) do
              D[i, j] := C[i, j]; {инициализация D}
    (2)   for j := 1 to (m + n –1) do
            begin
              (выбор из множества V\S такой
              вершины w, что значение D[i, w]
              минимально);
              добавить w к множеству S;
              for (каждая вершина v из
                  множества V\S) do
    D[i,v]:=min(D[i,v],D([i,w]+C[w,v]);
    (3)     end
    (4)   end
    end; { New}
```

Здесь массив *D* представляет собой результирующую матрицу кратчайших расстояний, причем на каждом шаге элемент *D[i, v]* содержит длину текущего кратчайшего пути от вершины *i* к вершине *v*. Массив *C* задает стоимости перелетов, где элемент *C[i, j]* равен стоимости дуги *i—>j*. Если дуги *i—>j* не существует, то *C[i, j]* полагается равным ∞ (бесконечности). Множество S имеет тот же смысл, что и в алгоритме Дейкстры.

Внешний цикл (строки 1–4) осуществляет последовательный перебор всех а\п вылетов, а внутренний цикл(строки 2 – 3) находит кратчайшие маршруты от этих а\п ко всем остальным, причем если в этом маршруте присутствуют промежуточные вершины, то они запоминаются.

Пример 3. В таблицах 4 и 5, соответственно, приведены матрица C и D, полученные с помощью нового алгоритма для орграфа, изображенного на рис. 1. В качестве а\п вылетов выбраны 1 и 2 (m=2).

Таблица 4. Матрица стоимостей

		А/п вылетов и посадок				
		1	2	3	4	5
А/п вылетов и посадок	11	∞	**10**	∞	**30**	**100**
	22	∞	∞	**50**	∞	∞
	13	**70**	∞	∞	∞	**10**
	24	∞	∞	**20**	∞	∞
	35	∞	∞	∞	**60**	∞

Таблица 5. Матрица кратчайших расстояний

		А/п вылетов и посадок				
		1	2	3	4	5
А\п вылетов	11	∞ ∞	**10**	**50**	**30**	**60**
	22	**120**	∞	**50**	**120**	**60**

Выводы по методу нахождения кратчайших маршрутов на транспортной сети

Новый алгоритм построения кратчайших путей между заданными множествами вершин в сети имеет следующие преимущества:

- полностью решает поставленную задачу, которая принципиально не могла быть решена в полном объеме с помощью алгоритма Дейкстры, в связи с неполным набором полученных им результатов. Это видно из последней строки четвертой итерации алгоритма Дейкстры, приведенного в табл. 2, которая является лишь частью матрицы кратчайших расстояний D, полученной с помощью нового алгоритма (см. табл. 5);
- более эффективно, т.е. проще и быстрее, решает задачу нахождения кратчайших путей между заданными множествами вершин в сети по сравнению, с хотя и адекватными, но избыточными результатами, которые получаются с использованием алгоритма Флойда. Об этом свидетельствует то, что матрица кратчайших расстояний D, найденная с помощью нового алгоритма (см. табл. 5), является лишь частью результата работы алгоритма Флойда, представленного на рис. 3 (первые две строки матрицы A_5);
- предложенный алгоритм, как в прочем и алгоритмы Дейкстры и Флойда, может использоваться при обработке сетевых моделей, задаваемых с помощью ориентированных и неориентированных графов.

Предложенный алгоритм построения кратчайших путей между заданными множествами вершин в сети реализован в виде программного комплекса, который прошел опытную апробацию при решении значительного количества сетевых транспортных задач. Результаты апробации доказали надежность и универсальность алгоритма, особенно на задачах большой размерности [6].

Определение ТЗ

Прежде чем остановиться на анализе методов решения открытых транспортных задач (ТЗ), дадим классическое определение ТЗ [7].

По определению ТЗ из m пунктов отправления $A_1,A_2,\ldots,A_m$ (ПО), в которых сосредоточены запасы однородного груза в количествах $a_1,a_2,\ldots,a_m$ единиц, необходимо перевезти этот груз в n пунктов назначения $B_1,B_2,\ldots,B_n$ (ПН) в соответствии с поступившими от них заявками на $b_1,b_2,\ldots,b_n$ единиц. Также предполагается,

что сумма всех заявок равняется сумме всех запасов, а именно:

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j . \quad (1)$$

Матрица (таблица) стоимостей перевозок единицы груза между каждым ПО и каждым ПН задается следующим образом:

c_{11}	c_{12}	...	c_{1n}
c_{21}	c_{22}	...	c_{2n}
...	...	...	...
c_{m1}	c_{m2}	...	c_{mn}

а матрица (таблица) соответствующих объемов перевозок имеет следующий вид:

x_{11}	x_{12}	...	x_{1n}
x_{21}	x_{22}	...	x_{2n}
...	...	...	...
x_{m1}	x_{m2}	...	x_{mn}

где c_{ij} – стоимость перевозки единицы груза из A_i в B_j, а x_{ij} – количество груза, перевозимое из A_i в B_j.

Перевезти груз нужно таким образом, чтобы все заявки были удовлетворены и при этом общая стоимость (Z) всех перевозок была бы минимальной, т.е.

$$Z = c_{11}x_{11}+c_{12}x_{12}+...+c_{mn}x_{mn} => min \quad (2)$$

ТЗ, отвечающая условию (1), называется закрытой или сбалансированной ТЗ [8].

Методы сведения открытых транспортных задачк сбалансированному виду

Существуют два наиболее известных метода сведения открытых ТЗ к сбалансированному виду [9]:

- ввод дополнительного (фиктивного) пункта отправления (назначения) груза;
- уменьшение объема спроса (предложения) на величину несоответствия у одного из ПН (ПО).

Первый метод имеет два случая использования:

а) если $\sum_{i=1}^{m} a_i > \sum_{j=1}^{n} b_j$, то есть предложение превышает спрос. При этом потребность фиктивного ПН B_{n+1} составляет $b_{n+1} = \sum_{i=1}^{m} a_i - \sum_{j=1}^{n} b_j$;

б) если $\sum_{i=1}^{m} a_i < \sum_{j=1}^{n} b_j$ – то есть спрос превышает предложение. В этом случае запасы фиктивного ПО составляют $a_{m+1} = \sum_{j=1}^{n} b_j - \sum_{i=1}^{m} a_i$. При этом в матрице стоимостей появляется, соответственно, дополнительный столбец или строка, которые означают нулевые стоимости перевозок. ТЗ становится сбалансированной, но при этом частично или полностью не вывозится груз у отдельных ПО при вводе дополнительного ПН или также частично или полностью не удовлетворяются заявки на получение груза некоторыми ПН при вводе дополнительного ПО.

Используя второй метод, для приведения ТЗ к сбалансированному виду, модуль разности $\left| \sum_{j=1}^{n} b_j - \sum_{i=1}^{m} a_i \right|$ вычитается у ПН (при $\sum_{i=1}^{m} a_i < \sum_{j=1}^{n} b_j$), имеющего наибольшее значение спроса, либо у ПО (при $\sum_{i=1}^{m} a_i > \sum_{j=1}^{n} b_j$), имеющего наибольшее значение предложения. Этот метод не применим в том случае, когда это наибольшее значение спроса(предложения) меньше вычитаемого модуля разности $\left| \sum_{j=1}^{n} b_j - \sum_{i=1}^{m} a_i \right|$.

Применение первого из методов зачастую не удовлетворяет заявки отдельных заказчиков грузов, особенно расположенных на значительном удалении от ПО. При использовании второго метода в невыгодном положении также оказываются наиболее удаленные склады.

С целью снятия этих двух существенных недостатков, предлагается новый метод сведения открытой ТЗ к сбалансированному виду, а именно коэффициентный метод балансирования ТЗ.

Коэффициентный метод сведения открытых ТЗ к закрытому виду

В основе этого метода заложено пропорциональное объемам заявок (запасов) уменьшение спроса (предложения) грузов всех без исключения ПН или ПО.

В случае превышения запасов груза над его спросом, рассчитывается коэффициент уменьшения объемов предложения:

$$k = \frac{\sum_{i=1}^{m} a_i - \sum_{j=1}^{n} b_j}{\sum_{i=1}^{m} a_i} \qquad (3)$$

После чего объемы предложения грузов всех ПО ($i=1,m$) уменьшаются до величин $a_i^* = (1-k)*a_i$ и задача становится сбалансированной с теми же объемами заявок b_j ($j=1,n$) и той же размерностью ТЗ.

При превышении же спроса на груз над его предложением, рассчитывается соответствующий коэффициент уменьшения объемов заявок всех ПН ($j=1,n$) по формуле (4). Затем эти объемы заявок уменьшаются до величин $b_j^* = (1-k)*b_j$ и ТЗ становится сбалансированной с теми же объемами запасов a_i ($i=1,m$) и той же размерностью.

$$k = \frac{\sum_{j=1}^{n} b_j - \sum_{i=1}^{m} a_i}{\sum_{j=1}^{n} b_j} \qquad (4)$$

Сравнительный анализ методов решения открытых ТЗ

В качестве примера рассмотрим ТЗ в виде двух начальных а/п вылетов (A_1 и A_2), двух промежуточных а/п (C_1 и C_2) и пяти конечных а/п (B_1, B_2, B_3, B_4 и B_5) (см. орграф на рис.4). Также там на дугах заданы затраты на перевозку единицы груза на единицу расстояния между отдельными а/п и объёмы груза, которые необходимо вывести из начальных а/п в конечные. Необходимо минимизировать транспортные расходы с условием того, что груза требуется на 10 единиц больше, чем его имеется. Для сравнения покажем применение всех названых методов сведения ТЗ к закрытому виду.

Оптимизация перевозок в приведенной выше ТЗ будет иметь вид:

а) в случае применения первого метода (ввода фиктивного начального а/п, в качестве которого мы возьмем промежуточный а/п C_1):

				Конечные а/п				
				B_1	B_2	B_3	B_4	B_5
				Объёмы заявок (b_j)				
				20	30	30	35	15
Начальные а/п	A_1	Объёмы поставок (a_i)	100	1 **20**	2 **30**	4 **10**	5 **35**	3 **5**
	A_2		20	3	2	2 **20**	4	4
	A_ϕ		10	5	4	4	4	2 **10**
Транспортные затраты на перевозку								350

Примечание: Затемнёнными в таблице показаны оптимальные объемы перевозок, осуществляемые по самым экономически выгодным маршрутам, стоимостям перевозки по которым соответствуют цифры в верхнем правом углу каждой клетки таблицы.

Эти маршруты показаны на графе (см. орграф на рис.4) пунктирными линиями и получены с помощью нового метода нахождения кратчайших маршрутов на транспортной сети.

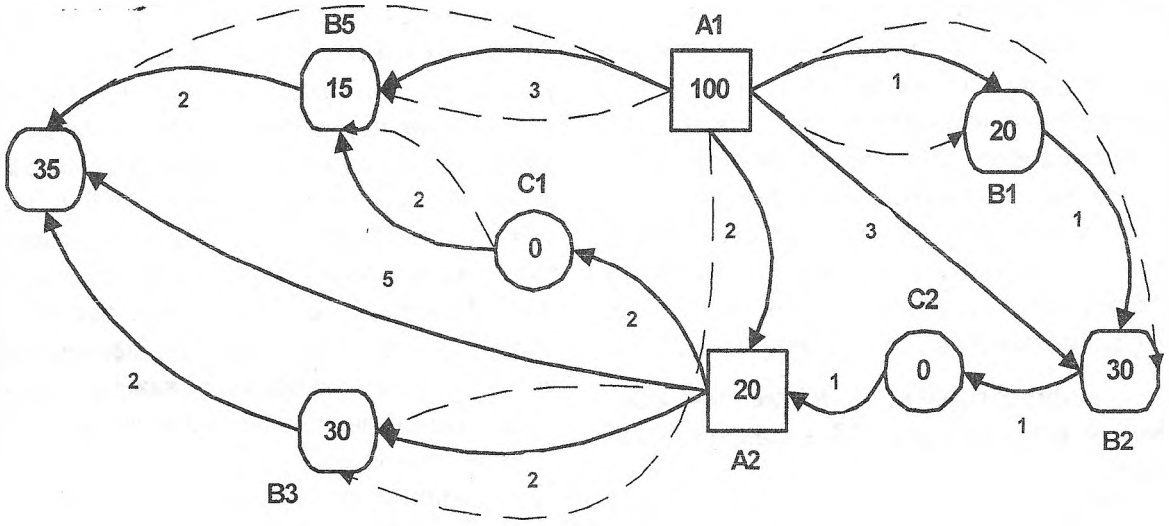

Рис. 4. ТЗ в виде орграфа (сети)

б) в случае применения второго метода (уменьшения объема самой большой заявки, а именно B_4, на величину несоответствия, т.е. 10):

				Конечные а/п				
				B_1	B_2	B_3	B_4	B_5
				Объёмы заявок (b_j)				
				20	30	30	25	15
Начальные а/п	A_1	Объёмы поставок(a_i)	100	1 20	2 30	4 10	5 25	3 15
	A_2		20	3	2	2 20	4	4
Транспортные затраты на перевозку							330	

в) в случае применения третьего метода (пропорционального уменьшения объема всех заявок):

				Конечные а/п				
				B_1	B_2	B_3	B_4	B_5
				Объёмы заявок (b_j)				
				18,46	27,69	27,69	32,31	13,85
Начальные а/п	A_1	Объёмы поставок(a_i)	100	1 18,46	2 27,69	4 7,69	5 32,31	3 13,85
	A_2		20	3	2	2 20	4	4
Транспортные затраты на перевозку							347,7	

И для сравнения приведем расчет транспортных перевозок при несбалансированном состоянии ТЗ:

				Конечные а/п				
				B_1	B_2	B_3	B_4	B_5
				Объёмы заявок (b_j)				
				20	30	30	35	15
Начальные а/п	A_1	Объёмы поставок(a_i)	100	1 16	2 25	4 22	5 22	3 15
	A_2		20	3 4	2 4,67	2 5,68	4 5,68	4
Транспортные затраты на перевозку							363,29	

В Национальном транспортном университете разработано программное обеспечение предлагаемого метода сведения ТЗ к сбалансированному виду. Его применение особенно эффективно при различии в расстояниях между отдельными ПО и ПН более чем в 2 раза.

Выводы по коэффициентному методу

Приведенные выше теоретические сведения о существующих методах сведения открытых ТЗ к сбалансированному виду и основанные на них экспериментальные расчеты оптимизации транспортных перевозок, полученные с помощью

разработанного программного комплекса, позволяют сделать следующие выводы:

во-первых, оптимизацию транспортных перевозок целесообразно производить только на сбалансированной ТЗ;

во-вторых, метод пропорционального уменьшения объемов спроса (предложения) груза у участников перевозочного процесса помимо его "справедливости", проявляемой при перераспределении объемов перевозок, и неограниченности применения, показывает относительно неплохой экономический результат;

в-третьих, большинство известных стандартных методов оптимизации транспортных перевозок (распределительный метод, метод потенциалов, метод дифференциальных рент и др.) вообще работают только с закрытыми ТЗ;

и в-четвертых, коэффициентный метод приведения открытых ТЗ к сбалансированному виду также применим и к сетевым ТЗ, являясь обязательным подготовительным (предварительным) этапом перед последующим применением процедур оптимизации транспортных грузовых перевозок на сетевых моделях транспортных систем.

В заключении хотелось бы отметить то, что совместное использование описанных в статье новых методов решения открытых сетевых транспортных задач, а именно: метода нахождения кратчайших маршрутов на транспортной сети и метода пропорционального перераспределения объемов перевозок между участниками перевозочного процесса, позволяет получить хороший экономический и организационно-технический эффекты.

Список литературы

1. *Прокудін Г. С., Білоус С. О.* Один з підходів до вирішення сітьової транспортної задачі // Безопасность дорожного движения на перекрестках Украины. – К.: ООО "Журнал "Радуга". – 2003. – № 1 – 2(15). – С. 52-56.

2. *Пападимитриу С., Стайглиц К.* Комбинаторная оптимизация. Алгоритмы и сложность. : Пер. с англ. – М.: Мир, 1985. – 325 с.

3. *Гудман С., Хидетниеми С.* Введение в разработку и анализ алгоритмов. – М.: Мир, 1981. – С. 309-320.

4. *Ахо А., Хопкрофт Д, Ульман Д.* Структуры данных и алгоритмы. : Пер. с англ. – М.: Издательский дом "Вильямс", 2001. – 384 с.

5. *Прокудін Г. С.* Модифікація методу Дейкстри стосовно розв'язання сітьових транспортних задач // Вісник НТУ, ТАУ. – К., 2002. – Вип. 7. – С. 195-198.

6. *Четверухін Б. М., Прокудін Г. С.* Моделі та алгоритми розв'язання сітьових транспортних задач великої розмірності // Автошляховик України // Окремий випуск // Вісник Північного наукового центру ТАУ. - К.: 2004.- № 7. – С. 11-15.

7. *Данциг Дж.* Линейное программирование, его применения и обобщения. – М.:"Прогресс", 1966. – 600 с.

8. *Зайченко Ю. П.* Исследование операций. – К.: Вища школа, 1979. – 392 с.

Четверухин Б. М. Исследование операций в транспортных системах: Учебное пособие. Часть I. Методы линейного программирования и их использование. – К.: УТУ, 2000. – 92 с.