TN 1506624 10/27

OCLC FirstSearch: Display

Your requested information from your library UNIVERSITAET GOETTINGEN

Return

SHIPPED - Lender

83131125

General Record Information

Request Identifier:	83131125	**Status:**	SHIPPED
Request Date:	20111018	**Source:**	ILLiad
OCLC Number:	310953383		
Borrower:	EYM	**Need Before:**	20111117
Receive Date:		**Renewal Request:**	
Due Date:	N/A	**New Due Date:**	
Lenders:	*ZXW, ZXW, ZXW		
Request Type:	Copy		
ALERT:			

Bibliographic Information

Call Number:	zb 17844:7
Title:	Vyc?islitel'naja matematika /
Imprint:	Moskva : Izd. Akad. Nauk, 1957-1961.
Article:	Lur'e, A.,: An algorithm for solving a transpotation problem by approximating by conditionally optimal plans
Volume:	7
Date:	1961
Pages:	151-160
Verified:	<TN:1506624><ODYSSEY:141.211.175.136/ILL> OCLC

My Library's Holdings Information

Borrowing Information

Patron:	Pellie, Seth
Ship To:	Interlibrary Loan/106 Hatcher Graduate Library/University of Michigan/913 South University Ave./Ann Arbor, MI 48109-1190/Ariel 141.211.175.21, ariel.lib.umich.edu
Bill To:	same
Ship Via:	Odyssey,UPS,FedEx,Rides for Michigan,USPS for Canada
Electronic Delivery:	Odyssey - 141.211.175.136/ILL
Maximum Cost:	IFM - $26.00
Copyright Compliance:	CCG
Fax:	(734) 936-3630 or (734) 647-2050
Email:	interlibrary.loan@umich.edu
Affiliation:	CIC, MRLT, SHARES, GMR Reciprocal

Lending Information

Lending Charges:	IFM - $15
Shipped:	20111021
Ship Insurance:	
Lending Notes:	Copies sent by FAX.
Lending Restrictions:	
Return To:	N/A
Return Via:	Library Rate

Return

А. Л. Лурье

АЛГОРИТМ РЕШЕНИЯ ТРАНСПОРТНОЙ ЗАДАЧИ ПУТЕМ ПРИБЛИЖЕНИЯ УСЛОВНО ОПТИМАЛЬНЫМИ ПЛАНАМИ

I. Введение. Сущность метода

В литературе по линейному программированию название транспортной получила следующая задача[1].

Требуется найти значения x_{ik} ($i = 1, 2, \ldots n$; $k = 1, 2, \ldots, m$), удовлетворяющие условиям:

$$\sum_k x_{ik} = a_i, \tag{1}$$

$$\sum_i x_{ik} = b_k, \tag{2}$$

$$x_ik \geqslant 0, \tag{3}$$

$$\sum_{i,k} c_{ik} x_{ik} = \min, \tag{4}$$

где a_i, b_k, c_{ik} — заданные действительные числа, причем $a_i > 0$, $b_k > 0$, $\Sigma a_i = \Sigma b_k$.

Одним из путей решения задач линейного программирования, предложенным Л. В. Канторовичем в 1939 г. [3], является «приближение условно оптимальными планами» [4]. В настоящей работе излагается соответствующий алгоритм для транспортной задачи[2].

---

[1] Эта задача была поставлена и разрешена применительно к планированию перевозок на заданной транспортной сети методом «круговых зависимостей» в советской плановой литературе в 1930 г. [1—2].

[2] В американской литературе сходные приемы решения транспортной задачи предложены Фордом и Фулкерсоном [5] и Манкресом [6] под названием «венгерского метода» (название указывает на связь предложений американских авторов с работами венгерских математиков по теории графов).

Сущность предлагаемого метода вкратце сводится к следующему. Подбираем систему значений x_{ik}, удовлетворяющую следующим условиям[1]:

$$x_{i_r,\ k_p} = 0, \text{ если } c_{i_r k_p} > \min_i c_{ik_p} \tag{5}$$

$$\sum_k x_{ik} \leqslant a_i. \tag{6}$$

$$\sum_i x_{ik} \leqslant b_k, \tag{7}$$

$$\sum_{i,k} x_{ik} = \max. \tag{8}$$

После этого определяем величины $N_k = b_k - \sum_i x_{ik}$. Если любое $N_k = 0$, задача решена. При $N_k > 0$ разбиваем значения i на два класса i' и i'' следующим образом.

К классу i'' относим любое i_r, если существует такое значение k, при котором

$$c_{i_r k} = \min_i c_{ik} \text{ и } N_k > 0.$$

К этому же классу присоединяем любое i_r, если существуют такие значения i'' и k, для которых $c_{i_r k} = \min_i c_{ik}$ и $x_{i''k} > 0$. Остальные i относим к классу i'.

Определяем

$$d = \min_{i',k'} (c_{i'k'} - \min_i c_{ik'}),$$

где под k' подразумеваются такие значения k, для которых выражение, стоящее в скобках, положительно при любых i'.

Каждое c_{ik} увеличиваем на d. Получив измененную систему значений c_{ik}, вновь начинаем проводить указанные выше операции.

При малых n и m применение изложенного метода весьма просто[2]. С увеличением n и m при подборе значений x_{ik}, удовлетворяющих условиям (5)—(8), возникают трудности, для разрешения которых необходима разработка специальной методики, и быстро возрастает число необходимых для решения операций. Для уменьшения этого числа желательно на каждой стадии решения задачи всзможно более полно использовать результаты операций, произведенных на предшествующих стадиях. В связи с этим приходится идти на некоторое усложнение алгоритма.

---

[1] Эта система значений x_{ik} и будет «условно оптимальным планом».

[2] См. краткое описание решения конкретного примера в работе [7] и более подробное в работе [8].

II. АЛГОРИТМ РЕШЕНИЯ ЗАДАЧИ

§ 1. Начальные операции и некоторые определения

Каждой строке матрицы $[c_{ik}]$ будем ставить в соответствие числа M_i, а каждому столбцу — числа N_k. Первоначально принимается $M_i = a_i$, $N_k = b_k$. Строки, для которых $M_i > 0$, будем называть абсолютно избыточными, столбцы, для которых $N_k > 0$, — неудовлетворенными, а остальные столбцы — удовлетворенными.

Для каждого столбца k матрицы $[c_{ik}]$ определяем его наименьшие элементы $c_{i_r k} = \min\limits_i c_{ik}$.

Строка i_r и столбец k_p считаются связанными друг с другом, если $c_{i_r k_p} = \min\limits_i c_{ik_p}$.

Будем говорить, что строка i_r снабжает столбец k_p, если, после того как установлена некоторая система значений переменных x_{ik}, имеет место неравенство $x_{i_r k_p} > 0$.

Некоторым строкам будем в дальнейшем ставить в соответствие цепи, т. е. ряды чисел вида $i_0 k_0; i_1 k_1; \ldots; i_s k_s$, где i_0 — номер строки, которой приписывается цепь, k_0 — номер связанного с ней столбца, i_1 — номер строки, снабжающей столбец k_0, k_1 — номер столбца, связанного со строкой i_1, и т. д. Строке, цепь которой состоит из $2(s+1)$ чисел ($s = 0, 1, 2, \ldots$), присваивается ранг s.

Устанавливаем значения переменных x_{ik}. Принимаем $x_{ik} = 0$, если строка i и столбец k не связаны друг с другом. Значения переменных x_{ik}, соответствующих пересечениям строк и столбцов, связанных друг с другом, определяем поочередно следующим образом. Полагаем каждое x_{ik} равным наименьшему из чисел M_i и N_k и одновременно уменьшаем оба эти числа на величину наименьшего из них [1].

Если после того, как x_{ik} определены, все $N_k = 0$, задача решена. При $N_k > 0$ просматриваем поочередно каждый неудовлетворительный столбец и устанавливаем, имеются ли связанные с ним строки, которым еще не приписаны цепи (цепь могла быть приписана при просмотре предшествующего столбца). Обнаружив такую строку, присваиваем ей цепь $i_0 k_0$. Строки, получившие цепи, назовем недостаточными, а все остальные — избыточными.

Переходим к § 2, считая при этом установленные в результате предыдущих операций недостаточные строки непросмотренными, т. е. подлежащими операциям, изложенным в § 2.

---

[1] Число последующих операций окажется, как правило, меньше, если при определении значений x_{ik}, соответствующих пересечениям связанных друг с другом строк и столбцов, придерживаться следующего порядка. Определяем сначала значения x_{ik} для тех строк, по которым $M_i \geqslant \Sigma N_{k_i}$, где k_i — номера столбцов, связанных со строкой i. Затем находим значения x_{ik} поочередно для каждой из остальных строк, начиная со строк, связанных с наименьшим числом столбцов. При этом по каждой строке начинаем с x_{ik}, соответствующих столбцам, связанным с наименьшим числом строк.

§ 2. Просмотр недостаточных строк

Начинаем просмотр с одной из непросмотренных строк, имеющих наименьший ранг, т. е. с одной из строк, которым приписаны наиболее короткие цепи. Если непросмотренных строк не оказывается, переходим к § 5.

Устанавливаем, снабжает ли просматриваемая строка (обозначим ее номер через i_1) какой-либо столбец, одновременно связанный с абсолютно избыточной строкой. Если такой столбец обнаружен, переходим к § 3. При отсутствии столбца, удовлетворяющего сформулированному выше условию, устанавливаем, снабжает ли строка i_1 какой-либо столбец, связанный с избыточной строкой, не являющейся абсолютно избыточной. Обозначим номер такого столбца (если он обнаружен) через k_0. Приписываем каждой избыточной строке, связанной со столбцом k_0, цепь $i_0k_0i_1k_1 \ldots i_sk_s$, где i_0 — номер данной избыточной строки, а $i_1k_1\, i_2k_2 \ldots \ldots i_sk_s$ — цепь просматриваемой строки i_1. Если строке i_0 была присвоена цепь при предыдущих операциях, то эта цепь считается утратившей силу. Переименовываем избыточные строки, получившие цепи, в недостаточные. Возвращаемся к началу § 2, считая при этом как переименованные строки, так и строку i_1 непросмотренными.

Если оказывается, что строка i_1 не снабжает столбцов, связанных с какими-либо избыточными строками, то признаем строку i_1 просмотренной и возвращаемся к началу § 2.

§ 3. Изменение значений переменных

Пусть $i_1k_1i_2k_2 \ldots i_sk_s$ — цепь недостаточной строки i_1, снабжающей столбец k_0, связанный с абсолютно избыточной строкой i_0. Определяем величину $d_s = \min (M_{i_0}, x_{i_1k_0}, x_{i_2k_1}, \ldots, x_{i_sk_{s-1}}, N_{k_s})$.

Уменьшаем на d_s величины $M_{i_0}, x_{i_1k_0}, x_{i_2k_1}, \ldots, x_{i_{s-1}k_{s-1}}, N_{k_s}$ и увеличиваем на d_s величины $x_{i_sk_0}, x_{i_1k_1}, \ldots, x_{i_sk_s}$.

Если крайним справа элементом ряда $M_{i_0}, x_{i_2k_0}, x_{i_2k_1}, \ldots, x_{i_sk_{s-1}}, N_{k_s}$, равным d_s, являлось M_{i_0}, то строке i_0 присваиваем цепь $i_0k_0i_1k_1 \ldots i_sk_s$ и переименовываем ее в недостаточную. Переходим к § 2, считая строку i_0 непросмотренной.

Если крайним справа элементом того же ряда, равным d_s, являлось $x_{i_rk_{r-1}}$ (r — одно из чисел 1, 2, 3, ..., s), то все строки, в цепи которых входит номер столбца k_{r-1}, переименовываем в избыточные. Если таких переименованных строк окажется меньше, чем недостаточных просмотренных строк, то переходим к § 4, считая при этом переименованные строки подлежащими проверке, т. е. операциям, изложенным в § 4. Если же переименованных строк не меньше, чем недостаточных просмотренных, переходим к § 2, причем недостаточные просмотренные строки вновь причисляем к непросмотренным.

Если имело место равенство $N_{k_s} = d_s$, а все остальные $N_k = 0$, то после операций вычитания и прибавления d_s задача оказывается решенной.

При наличии $N_k > 0$ все строки, в цепи которых входит номер столбца k_s, переименовываем в избыточные и затем поступаем точно так же, как в случае, рассмотренном в предыдущем абзаце.

§ 4. Проверка избыточных строк

Начинаем проверку с одной из подлежащих проверке строк, имеющих наименьший ранг. Если таких строк не оказывается, переходим к § 2. Обозначаем номер проверяемой строки через i_0, а ее ранг — через t.

Устанавливаем, имеет ли проверяемая строка i_0 связь со столбцом, снабжаемым недостаточной просмотренной строкой ранга $t-1$, или, при $t=0$, с неудовлетворенным столбцом. Если такого столбца не оказывается, устанавливаем, имеет ли строка i_0 связь со столбцом, снабжаемым недостаточной просмотренной строкой ранга t. Если строка i_0 оказывается связанной с неудовлетворенным столбцом, ей присваивается цепь i_0k_0, где k_0 — номер неудовлетворенного столбца[1]; если строка i_0 оказывается связанной со столбцом, снабжаемым недостаточной строкой ранга $t-1$ или t, ей присваивается цепь $i_0k_0i_1k_1\ldots i_sk_s$, где k_0 — номер соответствующего столбца, а $i_1k_1i_2k_2\ldots i_sk_s$ — цепь строки i_1, снабжающей столбец k_0. Строку i_0 рассматриваем в дальнейшем как недостаточную непросмотренную строку. Возвращаемся к началу § 4.

Если столбца, удовлетворяющего одному из указанных выше условий, не оказывается, то отмечаем все те столбцы, которые связаны со строкой i_0 и снабжаются недостаточными просмотренными строками любого ранга (если такие столбцы имеются). Из числа недостаточных просмотренных строк, снабжающих отмеченные столбцы, выделяем одну из строк, имеющих наименьший ранг, и переименовываем ее в непросмотренную. Строку i_0 признаем проверенной и возвращаемся к началу § 4.

Если столбцов, связанных со строкой i_0 и снабжаемых недостаточными просмотренными строками, не оказывается, возвращаемся к началу § 4, считая при этом строку i_0 проверенной.

§ 5. Преобразование матрицы

Обозначим через k' номера неудовлетворенных столбцов и столбцов, снабжаемых недостаточными строками (т. е. номера столбцов, не связанных с избыточными строками), а через i' — номера избыточных строк. Для каждого столбца k' определяем

$$d_{k'} = \min_{i'} c_{i'k'} - \min_{i} c_{ik'}.$$

Пусть $d = \min_{k'} d_{k'}$. Увеличиваем на d элементы недостаточных строк матрицы $[c_{ik}]$ или, если избыточных строк меньше, чем недостаточных, уменьшаем на d элементы избыточных строк. Отмечаем изменения в связях

[1] Цепь, ранее присвоенная строке, считается утратившей силу.

между строками и столбцами, являющиеся результатом изменений в матрице $[c'_{ik}]$.

У столбцов k', для которых имело место равенство $d_{k'} = d$ (назовем такие столбцы критическими[1]) появляются новые связи: если k'_p — критический столбец, а i'_r — строка, для которой имело место равенство $c'_{i'_r k'_p} = \min_{i'} c'_{i' k'_p}$, то после преобразования матрицы $[c_{ik}]$ строка i'_r и столбец k'_p оказываются связанными друг с другом.

В то же время в отношении столбцов, связанных до преобразования матрицы $[c_{ik}]$ как с недостаточными, так и с избыточными строками (если такие столбцы имелись), следует отметить прекращение связей с недостаточными строками.

§ 6. Дополнительные операции после преобразования матрицы

Устанавливаем, имеются ли критические неудовлетворенные столбцы, связанные с абсолютно избыточными строками.

Если обнаружен критический неудовлетворенный столбец (обозначим его через k_0), связанный с абсолютно избыточной строкой i_0, то увеличиваем значение соответствующей переменной $x_{i_0 k_0}$ на наименьшее из чисел M_{i_0} и N_{k_0}, одновременно уменьшая оба эти числа на величину наименьшего из них. Если при этом $M_{i_0} < N_{k_0}$, то возвращаемся к началу § 6. Если $M_{i_0} > N_{k_0}$, то все строки, в цепи которых входит номер столбца k_0, переименовываем в избыточные (и рассматриваем в дальнейшем как подлежащие проверке). После этого возвращаемся к началу § 6. Если $M_{i_0} = N_{k_0}$, то при условии, что все остальные $N_k = 0$, задача признается решенной; в противном случае поступаем так же, как при $M_{i_0} > N_{k_0}$.

При отсутствии критических неудовлетворенных столбцов, связанных с абсолютно избыточными строками, выясняем, имеются ли критические неудовлетворенные столбцы, связанные с избыточными строками, для которых $M_i = 0$. Если такой столбец обнаружен, то каждой избыточной строке, связанной с ним, приписываем цепь вида $i_0 k_0$, где i_0 — номер строки, k_0 — номер неудовлетворенного критического столбца[2], и считаем эти строки в дальнейшем недостаточными непросмотренными строками. Возвращаемся к началу § 6.

Если критических неудовлетворенных столбцов, связанных с избыточными строками, нет, то просматриваем критические удовлетворенные столбцы (если они имеются) и отмечаем недостаточные строки, снабжающие эти столбцы. Отмеченные строки (если такие оказались) считаем в дальнейшем непросмотренными. После этого, а также в случае, если критических удов-

[1] Наименование критических сохраняется за получившими его столбцами до окончания операций по § 6.

[2] Если данной строке была присвоена цепь при предыдущих операциях, то она считается утратившей силу.

летворенных столбцов, снабжаемых недостаточными строками, не оказалось, поступаем следующим образом. Если имеются подлежащие проверке избыточные строки и их число меньше числа недостаточных просмотренных строк, переходим к § 4. Если подлежащих проверке избыточных строк нет или они имеются, но их число не меньше числа недостаточных просмотренных строк, переходим к § 2, причем в последнем случае недостаточные просмотренные строки вновь причисляем к непросмотренным.

III. ОБОСНОВАНИЕ АЛГОРИТМА

Покажем, что после конечного числа операций алгоритм проводит к решению рассматриваемой задачи.

1. Перед каждым преобразованием матрицы $[c_{ik}]$ значения x_{ik} удовлетворяют условиям (5) — (8).

Путем непосредственного рассмотрения алгоритма легко убедиться в соблюдении условий (5) — (7) на любой стадии решения задачи, а также в том, что перед каждым переходом к § 5 нет ни одного неудовлетворенного столбца или столбца, снабжаемого недостаточной строкой, которые были бы связаны с избыточной строкой[1]. Из последнего следует, что снабжение неудовлетворенных столбцов и столбцов, снабжаемых недостаточными строками (т. е. Σx_{ik}, при обозначениях, принятых в § 5), не может быть увеличено за счет избыточных строк, иначе было бы нарушено условие (5). Отсюда следует, что так как для всех недостаточных строк $M_l = 0$, а все столбцы, снабжаемые избыточными строками, являются удовлетворенными, то Σx_{ik} не может быть увеличена, т. е. условие (8) действительно соблюдается перед каждым преобразованием матрицы $[c_{ik}]$.

2. После конечного числа операций алгоритм приводит к таким значениям переменных x_{ik} и элементов матрицы $[c_{ik}]$, при которых соблюдаются условия (1), (2) и (5).

Рассмотрение § 1—4 и 6 показывает, что каждый раз после преобразования матрицы $[c_{ik}]$ (для чего требуется конечное число операций, предусмотренных в § 5), а также на начальной стадии решения задачи конечное число операций приводит или к осуществлению условий (1) и (2) (для этого достаточно, чтобы для любого столбца $N_k = 0$), или к применению § 5 (т. е. к очередному преобразованию матрицы $[c_{ik}]$).

Следовательно, остается доказать, что число преобразований матрицы $[c_{ik}]$ конечно[2].

Матрицы $[c'_{ik}]$ и $[c''_{ik}]$ будем называть равносильными, если равенство

$$c'_{l_p k_p} = \min_i c'_{i k_p}$$

---

[1] К обнаружению и устранению таких «неправильных» столбцов сводятся, как нетрудно заметить, все операции в § 1—4 и 6. Результатом же применения § 5 является появление новых неправильных столбцов.

[2] Условие (5) соблюдается, как упомянуто выше, на любой стадии применения алгоритма.

является необходимым и достаточным условием равенства

$$c''_{i_r k_p} = \min_i c''_{ik_p}.$$

Если матрицы равносильны, то условия (5) — (8) удовлетворяются при одних и тех же значениях x_{ik}.

Преобразования матриц $[c_{ik}]$, осуществляемые при применении алгоритма, не могут привести к появлению матриц, равносильных ранее полученным. В самом деле, если такое преобразование влечет за собой изменение значений x_{ik}, а следовательно, и увеличение $\sum_{i,k} x_{ik}$, соответствующее условиям (5) — (8), то преобразованная матрица не может быть равносильной какой-либо из ранее полученных, поскольку на любой предшествующей стадии применения алгоритма $\sum_{i,k} x_{ik}$ была меньше, а следовательно, значения x_{ik} были другими.

Можно показать, что и в том случае, когда преобразования матрицы $[c_{ik}]$ не вызывают изменений в значениях x_{ik}, также не могут появиться равносильные матрицы.

При любом преобразовании матрицы $[c_{ik}]$ хотя бы один из ее элементов, не являвшийся минимальным в соответствующем столбце, становится таковым, т. е. для него оказывается справедливым равенство $c_{i_r k_p} = \min_i c_{ik_p}$.

Для преобразований матрицы $[c_{ik}]$, не вызывающих изменений в значениях x_{ik}, справедливо следующее положение: элемент матрицы, превратившийся в результате такого преобразования из неминимального в минимальный, может в дальнейшем вновь оказаться неминимальным лишь после того, как имело место хотя бы одно преобразование, связанное с изменениями значений x_{ik}. Отсюда следует, что при преобразованиях, не связанных с изменениями значений x_{ik}, не могут появиться равносильные матрицы.

Сформулированное выше положение в свою очередь легко можно доказать, исходя из следующих свойств рассматриваемого алгоритма:

а) при преобразовании матрицы $[c_{ik}]$ новый минимальный элемент может появиться лишь в столбце, который не снабжался избыточными строками;

б) при преобразовании матрицы $[c_{ik}]$ минимальный элемент может превратиться в неминимальный лишь в столбце, который снабжался избыточными строками;

в) преобразования матрицы $[c_{ik}]$, не вызывающие изменений в значениях x_{ik}, не могут привести к превращению какой-либо недостаточной строки в избыточную [1].

Для завершения доказательства положения п. 2 остается отметить, что число неравносильных матриц при заданных n и m конечно. Отсюда следует, что и число преобразований матриц также конечно, так как они не

[1] В этом случае происходят лишь превращения некоторых избыточных строк в недостаточные.

могут привести к появлению равносильных матриц. Тем самым положение п. 2 доказано.

Замечание. Выше через c_{ik} обозначались как элементы первоначальной матрицы $[c_{ik}]$, так и элементы преобразованных матриц. Если под c_{ik} понимать только элементы первоначальной матрицы (числа, заданные условиями задачи), а через c^0_{ik} обозначить элементы матрицы, полученной в результате осуществления алгоритма, то $[c^0_{ik}] = [c_{ik} + \lambda_i]$, где λ_i — некоторые действительные числа (определяемые операциями сложения и вычитания, предусмотренными в § 5). Условие (5) выразится при этом следующим образом:

$$x_{i_r k_p} = 0, \text{ если } c_{i_r k_p} + \lambda_{i_r} > \min_i [c_{ik_p} + \lambda_i]. \tag{5а}$$

3. $\sum_{i,k} c_{ik} x_{ik}$ принимает наименьшее значение, возможное при соблюдении условий (1) — (3), в том и только в том случае, если выполнено условие (5а) [1].

Примем для системы значений x_{ik}, удовлетворяющей требованиям (1)—(3) и условию (5а), обозначения x'_{ik}, а для системы значений x_{ik}, удовлетворяющей только условиям (1) — (3),— обозначения x''_{ik}. Поскольку имеется хотя бы одно значение x''_{ik}, для которого условие (5а) нарушено, очевидно, что

$$\sum_{i,k} (c_{ik} + \lambda_i) x'_{ik} = \sum_{i,k} c_{ik} x'_{ik} + \sum_{i,k} \lambda_i x'_{ik} <$$
$$< \sum_{i,k} (c_{ik} + \lambda_i) x''_{ik} = \sum_{i,k} c_{ik} x''_{ik} + \sum_{i,k} \lambda_i x''_{ik}, \tag{9}$$

но в силу условия (1)

$$\sum_k x'_{ik} = \sum_k x''_{ik} = a_i. \tag{10}$$

Отсюда

$$\sum_{i,k} \lambda_i x'_{ik} = \sum_{i,k} \lambda_i x''_{ik} = \sum_i \lambda_i a_i. \tag{11}$$

Из (9) и (11) получаем

$$\sum_{i,k} c_{ik} x'_{ik} < \sum_{i,k} c_{ik} x''_{ik}. \tag{12}$$

Поскольку при осуществлении операций, предусмотренных алгоритмом, требование (3) всегда соблюдается, из положений 2 и 3 вытекает, что после конечного числа операций алгоритм действительно приводит к решению задачи.

---

[1] Положение § 3 может рассматриваться как следствие теоремы о потенциалах Л. В. Канторовича [9]. В самом деле, λ_i являются потенциалами строк, а наименьшие для каждого столбца суммы $c_{ik} + \lambda_i$ — потенциалами соответствующих столбцов.

ЛИТЕРАТУРА

1. А. Н. Толстой. Методы нахождения наименьшего суммового километража при планировании перевозок. Сб. Планирование перевозок. М., Транспечать НКПС, 1930.
2. А. Н. Толстой. Методы устранения нерациональных перевозок при составлении оперативных планов. М., Трансжелдориздат, 1941.
3. Л. В. Канторович. Математические методы организации и планирования производства. Л., изд. ЛГУ, 1939.
4. Л. В. Канторович. О методах анализа некоторых экстремальных планово-производственных задач. Докл. АН СССР, 115, № 3, 1957.
5. L. R. Ford and D. R. Fulkerson. A simple algorithm for finding maximal network flows and an application to the Hitchcoock problem. Rand Corporation, Santa Monica, 1955, 743.
6. J. Muncres. Algorithms for the assignement and transportation problems. Journal of the society for industrial and applied mathematics, 5, N 1, 1957.
7. А. Александров, А. Лурье, Ю. Олейник. Применение электрочных вычислительных машин в оперативном планировании. Автомобильный транспорт, № 6, 1959.
8. А. Л. Лурье. Методы достижения наименьшего пробега грузов при составлении перевозочных схем. Сб. Применение математики в экономических исследованиях. М., Соцэкгиз, 1959.
9. Л. В. Канторович. О перемещении масс. Докл. АН СССР, 37, № 7—8, 1942.

АКАДЕМИЯ НАУК СССР

ВЫЧИСЛИТЕЛЬНЫЙ ЦЕНТР

ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА

Сборник 7

ИЗДАТЕЛЬСТВО АКАДЕМИИ НАУК СССР
Москва 1961

****
*** RX REPORT ***
****

RECEPTION OK

TX/RX NO	6798
DESTINATION TEL #	+49 551 395014
DESTINATION ID	
ST. TIME	10/21 10:05
TIME USE	02'15
PGS.	12
RESULT	OK