HOKKAIDO UNIVERSITY

Title	容量制約付き輸送問題に対する主・双対法のコードの作成
Author(s)	有坂, 英明; 大内, 東; 加地, 郁夫
Citation	北海道大學工學部研究報告, 89, 63-75
Issue Date	1978-11-02
Doc URL	https://hdl.handle.net/2115/41495
Type	departmental bulletin paper
File Information	89_63-76.pdf

Hokkaido University Collection of Scholarly and Academic Papers : HUSCAP

北海道大学工学部研究報告
第89号 (昭和53年)

Bulletin of the Faculty of Engineering,
Hokkaido University, No. 89 (1978)

容量制約付き輸送問題に対する主・双対法のコードの作成

有坂英明* 大内 東** 加地郁夫***

(昭和53年3月31日受理)

Primal-Dual Programming Code for Capacitated Transportation Problems

Hideaki ARISAKA Azuma O-UCHI Ikuo KAJI

(Received March 31, 1978)

Abstract

In this report, the primal-dual programming code for the capacitated transportation problem (algorithm, specification and programme) is represented. The programme code is written by FACOM FORTRAN Language for a FACOM-230-75 computer of the Hokkaido university computing center. Some numerical examples and results are described.

1. まえがき

輸送問題は，線形計画問題の特別の型で数多くの工学的・経済学的・経営学的な問題を表現する。輸送問題は，一般の線形計画問題を解くのに利用されているシンプレックス法を使用しても解くことができるが，膨大な記憶容量を必要とし，かつ大幅な計算時間を要し効率が悪い。そこで，問題の数学的構造を利用した効率的な解法が数多く考えだされている。著者らは輸送問題の解法として，ネットワークにおける最大流量を求める効率的なアルゴリズムであるラベリング法に基づいたプライマル・デュアル法による計算コードを作成したので報告する。

2. 輸送問題の定式化

ある同じ生産物が m 個の供給地に，それぞれ a_i 量 $(i=1, \cdots, m)$ ずつおいてある。それらは n 個の需要地に，それぞれ b_j 量 $(j=1, \cdots, n)$ ずつ送られる。供給地 i から需要地 j へ1単位送るための費用は c_{ij} である。また，供給地 i から需要地 j への輸送量 x_{ij} は，下限 l_{ij}，上限 d_{ij} に制約される。このとき問題は，総輸送費が最小になるように輸送量 x_{ij} を決めることである。したがって，輸送問題はつぎのような形で数学的に定式化される。

$$\text{(主問題)} \qquad \min\ Z=\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij}$$

$$\text{sub. to}$$

* 富士通株式会社
** 北海道大学工学部電気系統工学講座 助手
*** 北海道大学工学部電気工学科系統工学講座 教授

$$\sum_{j=1}^{n} x_{ij} = a_i \qquad (i=1,\cdots,m) \qquad \text{(供給制約)}$$

$$\sum_{i=1}^{m} x_{ij} = b_j \qquad (j=1,\cdots,n) \qquad \text{(需要制約)}$$

$$l_{ij} \leqq x_{ij} \leqq d_{ij} \quad (i=1,\cdots,m\,;\ j=1,\cdots,n) \quad \text{(容量制約)}$$

3. プライマル・デュアル法

プライマル・デュアル法は，輸送問題をネットワークの最大流量問題に帰着して解く方法である。これは，次のようにして考えることができる。一般の線形計画問題に対してプライマル・デュアル法を適用する場合，最初の段階では主基底には人為変数だけが含まれているだけで修正してゆくうちに人為変数がすべてゼロになったとき，主問題の実行可能基底解は最適解になる。輸送問題に対しては人為変数を明示的に導入する必要はない。なぜなら，人為変数は非負でなければならないから制約式は正規の変数だけで次のように表わすことができる。

$$\sum_{j=1}^{n} x_{ij} \leqq a_i \qquad (i=1,\cdots,m)$$

$$\sum_{i=1}^{m} x_{ij} \leqq b_j \qquad (j=1,\cdots,n)$$

このとき最適解は，人為変数の和を s とすると

$$\begin{aligned} s &= \sum_{i=1}^{m}\left(a_i - \sum_{j=1}^{n} x_{ij}\right) + \sum_{j=1}^{n}\left(b_j - \sum_{i=1}^{m} x_{ij}\right) \\ &= \sum_{i=1}^{m} a_i - \sum_{j=1}^{n} b_j - 2\sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij} \end{aligned}$$

を最小にする，すなわち $\sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij}$ を最大にする。したがって，輸送問題はネットワークの最大流量問題に帰着することができる。既述の主問題において，容量の下限 l_{ij} をゼロに変換して考えることができる。そこで，簡単化のため主問題の容量制約を $0 \leqq x_{ij} \leqq d_{ij}$ とすると双対問題は次のように表わされる。

$$\text{(双対問題)} \qquad \max z = \sum_{i=1}^{m} a_i u_i + \sum_{j=1}^{n} b_j v_j + \sum_{i=1}^{m}\sum_{j=1}^{n} d_{ij} w_{ij}$$

sub. to

$$u_i + v_j + w_{ij} \leqq c_{ij} \quad (i=1,\cdots,m\,;\ j=1,\cdots,n)$$

$$w_{ij} \leqq 0 \quad (i=1,\cdots,m\,;\ j=1,\cdots,n)$$

u_i, v_j は符号無制約

双対問題に対する初期実行可能解は容易に求められる。すなわち，u_i, v_j の値を次のようにとる。

$$u_i = \min_j c_{ij} \qquad (i=1,\cdots,m)$$

$$v_j = \min_i (c_{ij} - u_i) \quad (j=,n)$$

これらの変数は双対問題の制約条件を満たし，その解となるから，それに対応する主問題はすべての変数について $x_{ij}=0$ から開始する必要がなく反復回数の減少となる。

Table 1. Tableau form for primal-Dual Algorithm

		D_1		D_2		$\cdots$	D_n				
	v_j / u_i	v_1		v_2		$\cdots$	v_n		a_i / x_s^i	δ_i	γ_i
O_1	u_1	c_{11} / x_{11}	d_{11} / w_{11}	c_{12} / x_{12}	d_{12} / w_{12}	$\cdots$	c_{1n} / x_{1n}	d_{1n} / w_{1n}	a_1 / x_s^1	δ_1	γ_1
O_2	v_2	c_{21} / x_{21}	d_{21} / w_{21}	c_{22} / x_{22}	d_{22} / w_{22}	$\cdots$	c_{2n} / x_{2n}	d_{2n} / w_{2n}	a_2 / x_s^2	δ_2	γ_2
$\vdots$	$\vdots$	$\vdots$		$\vdots$			$\vdots$		$\vdots$	$\vdots$	$\vdots$
O_m	u_m	c_{m1} / x_{m1}	d_{m1} / w_{m1}	c_{m2} / x_{m2}	d_{m2} / w_{m2}	$\cdots$	c_{mn} / x_{mn}	d_{mn} / w_{mn}	a_m / x_s^m	δ_m	γ_m
	b_j / x_{sj}	b_1 / x_{s1}		b_2 / x_{s2}		$\cdots$	b_n / x_{sn}		$\sum x_s^i$		
	ξ_j	ξ_1		ξ_2		$\cdots$	ξ_n			Lables	
	ρ_j	ρ_1		ρ_2		$\cdots$	ρ_n				

$$x_s^i = a_i - \sum_{j=1}^{n} x_{ij},\ x_{sj} = b_j - \sum_{i=1}^{m} x_{ij}$$

2.1 アルゴリズム

プライマル・デュアル法では Table 1 のタブローを使用する。以下にアルゴリズムを記述する。アルゴリズムの概略は第1図に示してある。

Step 1. フロー可能な輸送経路の選定（双対問題の初期解の決定）

限定主問題で正になりえる x_{ij} は○でかこんだ要素である。

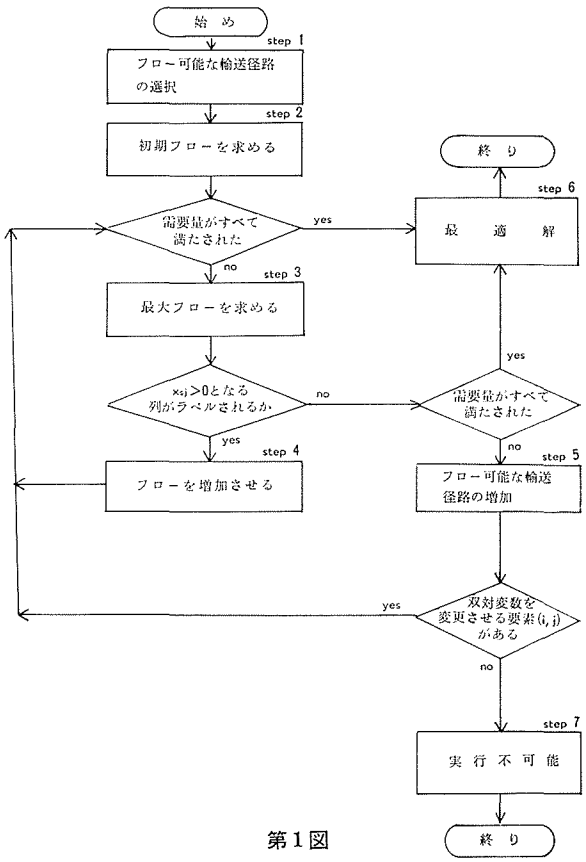
(イ) 単位輸送コストの値をテーブルに記入する。($i=1, \cdots, m$; $j=1, \cdots, n$)

(ロ) 各行について，$u_i = \min_j c_{ij}$ ($i=1, \cdots, m$) を求め u_i 欄に記入する。

(ハ) $u_i = c_{ij}$ ($i=1, \cdots, m$) となる要素を○でかこむ。

(ニ) 1個あるいは，それ以上の○のある列に対して $v_j=0$ とおく。

(ホ) 1個も○のない列に対して，$\phi_{ij} = c_{ij} - u_i$ ($i=1, \cdots, m$) を計



第1図

算し，$v_j=\min_i \phi_{ij}$ を求め v_j 欄に記入する。

(ヘ) $v_j=\phi_{ij}$ となる要素を○でかこむ。

(ト) $w_{ij}=0$ とおく。

Step 2. 初期フローを求める

2-a 第1行から始める。丸印の要素を見つける。丸印の要素 $(1, j)$ があれば，$x_{1j}=\min(d_{1j}, a_1, b_j)$ とおく。$x_{1j}=b_j$ または $x_{1j}=d_{1j}$ すなわち $x_{1j}<a_1$ なら別の丸印の要素を見つけ，たとえば $(1, k)$ があったとすれば $x_{1k}=\min(d_{1k}, a_1-x_{1j}, b_k)$ とする。これは，$a_1-x_{1j}-x_{1k}-\cdots$ すなわち $a_1-\sum_{j=1}^{n} x_{1j}$ が○になるか1行目に丸印の要素がなくなるまで続ける。そこで，$a-\sum_j x_{1j}$ を計算しその値を x_{s1} とする。供給余裕は，$x_s^i=a_i-\sum_j x_{1j}$ で表わされる。

2-b 次に第2行に移る。最初の丸印の要素 $(2, r)$ に対して，$x_{2r}=\min(d_{2r}, a_{2r}, b_r-x_{1r})$ とおく。ここで，需要地 r における需要量 b_r のうち2-aで供給地1から x_{1r} だけ充足されているから需要地 r における未充足需要量は b_r-x_{1r} となっている。このようにして，最後の行まで続けてゆくと初期フローを得ることができる。なお各行で丸印がついてその要素の値が正である x_{ij} が求められ，$x_s^i=a_i-\sum_j x_{ij}$ (i 行目すなわち供給地 i における供給余裕量) が計算される。行の計算処理が終わると各列について，$x_{sj}=b_j-\sum_i x_{ij}$ (j 列目すなわち需要地 j における未充足需要量) が計算される。このようにして初期フローが求められる。

$\sum_{j=1}^{n} x_{sj}=0$ なら Step 6 に進む。

$\sum_{j=1}^{n} x_{sj}\neq 0$ なら Step 3 に進む。

Step 3. 最大フローを求める

$x_s^i>0$ である行 (すなわち x_s^i はソースから供給地 i を結ぶアークの供給余裕量である) に，$s_i=x_s^i$, $r_i=s$ (s はソースを表わす) とおくことによって行にラベルする。ラベルされた各行において丸印の要素を探す。これらの要素では必ずしも $x_{ij}>0$ である必要はない。このラベルされた行 i のなかで丸印の要素を含んでいる列 j に対して，$\varepsilon_j=\min(d_{ij}-x_{ij}, \delta_i)$, $\rho_j=i$ とおくことによって列にラベルする。この操作をラベルされた最小のインデックス i から始めラベルされたすべての行について行ない列にラベルをつけ終える。ここで，$x_{sj}>0$ となる列がラベルされているならフローを変更するために Step 4 に進む。$x_{sj}>0$ となる列が1つもラベルされていないなら以下の3-aに進む。

3-a ラベルされた列のなかで $x_{ij}>0$ となっている丸印の要素を探す。その要素がラベルされていない行 i にあれば，その行に $s_i=\min(x_{ij}, \xi_j)$, $r_j=j$ とおくことによりラベルする。新しくラベルされた行 (いくつかあるかもしれない) にそって，まだ列にはラベルされていないような丸印の要素を探し $\xi_j=\min(d_{ij}-x_{ij}, \delta_i)$, $\rho_j=i$ とおくことによってその列にラベルする。もし，$x_{sj}>0$ となる列がまだラベルされていないなら3-aにもどり，その手続きをくり返す。この操作は，次の2つのケースのいずれかで終る。

(i) $x_{sj}>0$ となる列をラベルすることができなくなる。

この場合には，フローは最大になる。$\xi=\sum_{j=1}^{n} x_{sj}$ を計算する。

$\xi=\sum_{j=1}^{n} x_{sj}=0$ なら最適解が得られたので Step 6 へ進む。

$\xi=\sum_{j=1}^{n} x_{sj}\neq 0$ なら現在の経路ではこれ以上フローを増加することはできないので双対変数を変更して経路をふやすため Step 5 へ進む。

(ii) $x_{sj}\geqq 0$ となる列がラベルされる。この場合フロー増加可能なパスに沿ってフローを増加させるため Step 4 に進む。

Step 4. フローを増加させる

k を $x_{sk}>0$ でラベルされた列であるとする。この場合には，フローを増加させることができる。新しいタブローがつくられる。まず，値の変わる x_{ij} を考える。$\sigma=\min(\xi_k, x_{sk})$ と表わすとき，もし $\rho_k=u$ なら $\bar{x}_{uk}=x_{uk}+\sigma$，もし $r_u=t$ なら $\bar{x}_{ut}=x_{ut}-\sigma$，もし $\rho_t=v$ なら $\bar{x}_{vt}=x_{ut}+\sigma$ というように，経路にある x_{ij} の値は交互に σ だけ増減する。それ以外の x_{ij} については，もとのままで $\bar{x}_{ij}=x_{ij}$ である。また，双対問題の初期解は新しいタブローを作るときにも変わらない。限定主問題を最適にするために Step 3 に戻る。

Step 5. フロー可能な輸送経路の増加

(双対問題の初期解の変更)

$x_{sj}>0$ となる列をもはやラベルすることができないならフローは最大である。(未充足需要量を残したまま与えられた経路ではそれ以上フローを増加することはできない) そこで，少なくとも新しい1つの x_{ij} が正になることができる要素を見つける。まず，$i\in I$ をラベルされた行，$i\notin I$ をラベルされていない行，$j\in J$ をラベルされた列，$j\notin J$ をラベルされていない列の集合とするときそれらを組み合わせた要素 (i, j) について，R を双対問題の制約式 $u_i+v_j+w_{ij}=c_{ij}$ の等号が成立するインデックス ij の集合とし，S を双対変数 $w_{ij}<0$ が成立するインデックス ij の集合とする。

$$h=\min_{\substack{i\in I\\ j\notin J\\ ij\notin R}}[\min(c_{ij}-u_i-v_j-w_{ij}),\ \min_{\substack{i\in I\\ j\in J\\ ij\in S}}|w_{ij}|]>0$$

とおくと，双対変数の新しい値 $\hat{u}_i$, $\hat{v}_j$, $\hat{w}_{ij}$ は，それぞれ次のように変化する。

$$\hat{u}_i=\begin{cases}u_i+h, & i\in I\\ u, & i\notin I;\end{cases}\qquad \hat{v}_j=\begin{cases}v_j-h, & j\in J\\ v_j, & j\notin J;\end{cases}$$

$$w_{ij}=\begin{cases}w_{ij}-h, & i\in I,\ j\notin J,\ \text{and } ij\in R,\\ w_{ij}+h, & i\in I,\ j\in J,\ \text{and } ij\in S,\\ w_{ij} & \text{all other } i,j\end{cases}$$

h の値を決めるインデックス ij の集合が存在しないときは，$h=\infty$ となる。このとき双対問題は非有界となり主問題は実行不可能である。この場合 Step 7 へ進む。新しく得られた双対変数 $\hat{u}_i$, $\hat{v}_j$, $\hat{w}_{ij}$ で u_i, v_j, w_{ij} を変更して新しいタブローを作る。そのタブローで，$u_i+v_j+w_{ij}=c_{ij}$ となる要素 (i, j) に丸印をつける。この場合前のタブローで $x_{ij}>0$ となっていた要素 (i, j) は，すべて新しいタブローで丸印を含む。また新しいタブローでは，前のタブローで丸印のつけられていない要素が少なくとも1つは丸印がつけられる。前のタブローから新しいタブローに x_{ij} の値を入れる。最大フローを与える解であるが，$\sum_{j=1}^{n}x_{sj}>0$ であるため与えられた主問題の最適解とはなっていない前の限定問題の最適解が新しい限定主問題の初期解として用いられる。Step 3 に戻る。

Step 6. 最適解の決定

限定主問題において $\sum_{j=1}^{n}x_{sj}=0$ となれば，与えられた主問題に対する実行可能解を得たことになるから最適解が得られたことになる。したがって，目的関数の値は $\min Z=\max z=\sum_{i=1}^{m}a_iu_i+\sum_{j=1}^{n}b_jv_j+\sum_{i=1}^{m}\sum_{j=1}^{n}d_{ij}w_{ij}$ より容易に計算することができる。

Step 7. 実行不可能

このとき，双対問題の目的関数は非有界となるから主問題は実行不可能となる。

(注意)

解が存在すれば，上述の計算過程は双対変数の新しい解が得られるごとに $\bar{Z}>Z$ となるから有限回の反復計算で終了する。

4. プログラムの構成および使用方法

4.1 プログラムの構成

プライマル・デュアル法のプログラムのおおよその流れ図を第1図に示す。本プログラムは，記憶容量を可変にできる整合寸法を用いているためメインプログラム作成に際しては注意を要する。

4.2 プログラムの使用方法

(1) 入力情報

M: 供給地点の個数

N: 需要地点の個数

IMPAX: 反復回数の上限値を与える。通常 $(M+N)$ の2倍程度の大きさにとる。

MP: タブローの行数 $(=M+3)$

NP: タブローの列数 $(=N+3)$

$A(i)$: $i=1, \cdots, M$ に供給地点 i での供給量を入れる。

$B(j)$: $j=1, \cdots, N$ に需要地点 j での需要量を入れる。

$L(i, j)$: $i=1, \cdots, M$; $j=1, \cdots, N$ に変数の下限を入れる。

$D(i, j)$: $i=1, \cdots, M$; $j=1, \cdots, N$ に変数の上限を入れる。

$C(i, j)$: $i=1, \cdots, M$; $j=1, \cdots, N$ に変数のコストを入れる。

(2) 出力情報

IOPT: 最適解のときは1，実行不可能のときは2，反復回数が IPMAX を越えたときは3が入る。

$X(i, j)$: $i=1, \cdots, M$; $j=1, \cdots, N$ に最適然の値が入る。

IZ: 得られた目的関数の値が入る。

(3) 必要な配列宣言

1次元配列	大きさ M:	A, U
	大きさ N:	B, V
2次元配列	大きさ $M\times N$:	L, D, C
	大きさ $MP\times NP$:	X

したがって必要な記憶容量は，約 $5mn+2(m+n)$ となる。

(注意)

1. 供給地点 i から需要地点 j 間の経路 (i, j) が存在しない場合は，変数の下限，上限を，それぞれゼロとおけばよい。その場合，コストは ∞ とする。
2. 本解法では，供給量の総和と需要量の総和が等しい，すなわち $\sum_{i=1}^{n} a_i = \sum_{j=1}^{n} b_j$ である必要はない。供給量の総和が需要量の総和より大きい，すなわち $\sum_{i=1}^{m} a_i \geqq \sum_{j=1}^{n} b_j$ のときもアルゴリズムを変更しないで使用できる。

5. FORTRAN プログラム

```
      SUBROUTINE CATRA(M,N,A,B,L,D,C,X,U,V,W,MP,NP,IPMAX,IOPT,IZ)
      INTEGER  A(M),B(N),D(M,N),C(M,N)
      DIMENSION L(M,N)
      INTEGER X(MP,NP),U(M),V(N),W(M,N)
      INTEGER CNT,P1,T5,S,U5
      M1=M+1
      N1=N+1
      CNT=0
      DO 10 J=N1,NP
      DO 10 I=1,MP
   10 X(I,J)=0
      DO 20 J=1,NP
      DO 20 I=M1,MP
   20 X(I,J)=0
      DO 30 J=1,N
      DO 30 I=1,M
      X(I,J)=-1
   30 W(I,J)=0
      DO 40 I=1,M
      X(I,N+1)=A(I)
   40 U(I)=0
      DO 50 J=1,N
      X(M+1,J)=B(J)
   50 V(J)=-1
      DO 60 J=1,N
      DO 60 I=1,M
   60 D(I,J)=D(I,J)-L(I,J)
      DO 70 I=1,M
      IS=0
      DO 80 J=1,N
      IS=IS+L(I,J)
   80 CONTINUE
      X(I,N+1)=X(I,N+1)-IS
   70 CONTINUE
      DO 90 J=1,N
      IS=0
      DO 95 I=1,M
      IS=IS+L(I,J)
   95 CONTINUE
      X(M+1,J)=X(M+1,J)-IS
   90 CONTINUE
C     ***** STEP 1 *****
      DO 100 I=1,M
      S=0
      U(I)=C(I,1)
      DO 110 J=1,N
      IF(S.NE.0) GO TO 120
      S=S+1
      GO TO 130
  120 IF(C(I,J).GE.U(I)) GO TO 110
  130 U(I)=C(I,J)
      J1=J
  110 CONTINUE
      DO 140 J=J1,N
      IF (C(I,J).NE.U(I)) GO TO 140
      X(I,J)=0
      V(J)=0
  140 CONTINUE
  100 CONTINUE
      DO 150 J=1,N
      IF(V(J).EQ.0) GO TO 150
      S=0
      V(J)=C(1,J)-U(1)
      DO 160 I=1,M
      IF(S.NE.0) GO TO 170
      S=S+1
      IE=C(I,J)-U(I)
      GO TO 180
```

```
  170 IE=C(I,J)-U(I)
      IF(IE.GE.V(J)) GO TO 160
  180 V(J)=IE
      I1=I
  160 CONTINUE
      DO 190 I=I1,M
      IF(V(J).NE.C(I,J)-U(I)) GO TO 190
      X(I,J)=0
  190 CONTINUE
  150 CONTINUE
C     ***** STEP 2 *****
      I=1
  200 J=1
  210 IF(X(M+1,J).EQ.0) GO TO 240
      IF(X(I,N+1).EQ.0) GO TO 280
      IF(X(I,J).NE.0) GO TO 240
      IF(X(I,N+1).LE.X(M+1,J)) GO TO 250
      IF(D(I,J).LT.X(M+1,J)) GO TO 220
      X(I,J)=X(M+1,J)
      GO TO 230
  220 X(I,J)=D(I,J)
  230 X(I,N+1)=X(I,N+1)-X(I,J)
      X(M+1,J)=X(M+1,J)-X(I,J)
  240 J=J+1
      IF(J.LE.N) GO TO 210
      IF(I.LE.M) GO TO 280
      GO TO 290
  250 IF(D(I,J).LT.X(I,N+1)) GO TO 260
      X(I,J)=X(I,N+1)
      GO TO 270
  260 X(I,J)=D(I,J)
  270 X(M+1,J)=X(M+1,J)-X(I,J)
      X(I,N+1)=X(I,N+1)-X(I,J)
      IF(X(I,N+1).NE.0) GO TO 240
  280 I=I+1
      IF(I.LE.M) GO TO 200
  290 S=0
      DO 295 J=1,N
  295 S=S+X(M+1,J)
      IF(S.EQ.0) GO TO 600
C     ***** STEP 3 *****
  300 CNT=CNT+1
      IF(IPMAX.EQ.CNT-1) GO TO 680
      DO 305 I=1,M
      IF(X(I,N+1).LE.0) GO TO 305
      X(I,N+2)=X(I,N+1)
      X(I,N+3)=-1
      DO 310 J=1,N
      IF(X(M+3,J).NE.0) GO TO 310
      IF(X(I,J).LT.0) GO TO 310
      IF(D(I,J).LE.X(I,J)) GO TO 310
      I5=D(I,J)-X(I,J)
      IF(I5.LT.X(I,N+2)) GO TO 315
      X(M+2,J)=X(I,N+2)
      GO TO 320
  315 X(M+2,J)=I5
  320 X(M+3,J)=I
  310 CONTINUE
  305 CONTINUE
      DO 325 K=1,N
      IF(X(M+3,K).EQ.0) GO TO 325
      IF(X(M+1,K).GT.0) GO TO 400
  325 CONTINUE
  330 J=1
  335 IF(X(M+3,J).EQ.0) GO TO 380
      S=0
      DO 340 I=1,M
      IF(X(I,J).LE.0) GO TO 340
      IF(X(I,N+3).NE.0) GO TO 340
      IF(X(I,J).NE.D(I,J)) GO TO 345
      IF(W(I,J).LT.0) GO TO 340
  345 IF(X(I,J).GE.X(M+2,J)) GO TO 350
      X(I,N+2)=X(I,J)
      GO TO 355
```

```
  350 X(I,N+2)=X(M+2,J)
  355 X(I,N+3)=J
      S=S+1
      DO 360 K=1,N
      IF(X(I,K).LT.0) GO TO 360
      IF(X(M+3,K).NE.0) GO TO 360
      IF(D(I,K).LE.X(I,K)) GO TO 360
      I5=D(I,K)-X(I,K)
      IF(X(I,N+2).LE.I5) GO TO 365
      X(M+2,K)=I5
      GO TO 370
  365 X(M+2,K)=X(I,N+2)
  370 X(M+3,K)=I
      IF(X(M+1,K).GT.0) GO TO 400
  360 CONTINUE
  340 CONTINUE
      IF(S.EQ.0) GO TO 380
      GO TO 330
  380 IF(J.EQ.N) GO TO 385
      J=J+1
      GO TO 335
  385 S=0
      DO 390 J=1,N
  390 S=S+X(M+1,J)
      IF(S.EQ.0) GO TO 600
      GO TO 500
C     ***** STEP 4 *****
  400 IF(X(M+2,K).GE.X(M+1,K)) GO TO 410
      P1=X(M+2,K)
      GO TO 420
  410 P1=X(M+1,K)
  420 U5=X(M+3,K)
      X(U5,K)=X(U5,K)+P1
      X(M+1,K)=X(M+1,K)-P1
      X(U5,N+1)=X(U5,N+1)-P1
      T5=X(U5,N+3)
      IF(T5.NE.-1) GO TO 430
      GO TO 597
  430 X(U5,T5)=X(U5,T5)-P1
      X(M+1,T5)=X(M+1,T5)+P1
      X(U5,N+1)=X(U5,N+1)+P1
      K=T5
      GO TO 420
C     ***** STEP 5 *****
  500 S=0
      MINH=0
      L1=0
      DO 505 I=1,M
      IF(X(I,N+3).EQ.0) GO TO 505
      DO 510 J=1,N
      IF(X(M+3,J).NE.0) GO TO 510
      IF(U(I)+V(J)+W(I,J).EQ.C(I,J)) GO TO 510
      JE=C(I,J)-U(I)-V(J)
      IF(S.NE.0) GO TO 515
      MINH=JE
      S=S+1
      GO TO 520
  515 IF(JE.GE.MINH) GO TO 510
  520 MINH=JE
      L1=L1+1
  510 CONTINUE
  505 CONTINUE
      S=0
      MIN=0
      L2=0
      DO 525 I=1,M
      IF(X(I,N+3).NE.0) GO TO 525
      DO 530 J=1,N
      IF(X(M+3,J).EQ.0) GO TO 530
      IF(W(I,J).GE.0) GO TO 530
      IF(S.NE.0) GO TO 535
      MIN=IABS(W(I,J))
      S=S+1
      GO TO 540
```

```
  535 IF(IABS(W(I,J)).GE.MIN) GO TO 530
  540 MIN=IABS(W(I,J))
      L2=L2+1
  530 CONTINUE
  525 CONTINUE

      IF(L1+L2.EQ.0) GO TO 670
      IF(MIN.EQ.0) GO TO 550
      IF(MINH.EQ.0) GO TO 545
      IF(MINH.LE.MIN) GO TO 550
  545 MINH=MIN
  550 DO 555 I=1,M
      IF(X(I,N+3).EQ.0) GO TO 565
      DO 560 J=1,N
      IF(X(M+3,J).NE.0) GO TO 560
      IF(U(I)+V(J)+W(I,J).NE.C(I,J)) GO TO 560
      W(I,J)=W(I,J)-MINH
  560 CONTINUE
      GO TO 555
  565 DO 570 J=1,N
      IF(X(M+3,J).EQ.0) GO TO 570
      IF(W(I,J).GE.0) GO TO 570
      W(I,J)=W(I,J)+MINH
  570 CONTINUE
  555 CONTINUE
      DO 580 I=1,M
      IF(X(I,N+3).EQ.0) GO TO 580
      U(I)=U(I)+MINH
  580 CONTINUE
      DO 585 J=1,N
      IF(X(M+3,J).EQ.0) GO TO 585
      V(J)=V(J)-MINH
  585 CONTINUE
      DO 590 I=1,M
      DO 595 J=1,N
      IF(X(I,J).GT.0) GO TO 595
      IF(U(I)+V(J)+W(I,J).NE.C(I,J)) GO TO 596
      X(I,J)=0
      GO TO 595
  596 X(I,J)=-1
  595 CONTINUE
  590 CONTINUE
  597 DO 598 I=1,M
      X(I,N+2)=0
  598 X(I,N+3)=0
      DO 599 J=1,N
      X(M+2,J)=0
  599 X(M+3,J)=0
      GO TO 300
C     ***** STEP 6 *****
  600 IOPT=1
      DO 610 J=1,N
      DO 620 I=1,M
      IF(X(I,J).NE.-1) GO TO 630
      X(I,J)=L(I,J)
      GO TO 620
  630 X(I,J)=X(I,J)+L(I,J)
  620 CONTINUE
  610 CONTINUE
      IZ=0
      DO 640 J=1,N
      DO 650 I=1,M
      IF(X(I,J).LE.0) GO TO 650
      IZ=IZ+C(I,J)*X(I,J)
  650 CONTINUE
  640 CONTINUE
      GO TO 660
C     ***** STEP 7 *****
  670 IOPT=2
      GO TO 660
  680 IOPT=3
  660 RETURN
      END
```

6. 使 用 例

6.1 メインプログラム

```
C     ***** MAIN PROGRAM *****
      INTEGER A(3),B(4),D(3,4),C(3,4)
      DIMENSION L(3,4)
      INTEGER X(6,7),U(3),V(4),W(3,4)
      WRITE(6,1)
    1 FORMAT(1H ,30H***** PRIMAL-DUAL METHOD *****)
      READ(5,1000) M,N
 1000 FORMAT(2I5)
      WRITE(6,5) M,N
    5 FORMAT(1H ,2HM=,I2,/,1H ,2HN=,I2)
      MP=M+3
      NP=N+3
      IPMAX=20
      IOPT=0
      DO 10 J=1,N
      DO 10 I=1,M
      READ(5,2000) L(I,J),D(I,J),C(I,J)
 2000 FORMAT(3I5)
   10 CONTINUE
      READ(5,3000) (A(I),I=1,M)
 3000 FORMAT(I5)
      READ(5,4000) (B(J),J=1,N)
 4000 FORMAT(I5)
      WRITE(6,100) M,N
  100 FORMAT(1H ,2HL(,I2,1H*,I2,1H))
      DO 110 I=1,M
  110 WRITE(6,800) (L(I,J),J=1,N)
  800 FORMAT(1H ,4(12I5))
      WRITE(6,200) M,N
  200 FORMAT(1H ,2HD(,I2,1H*,I2,1H))
      DO 210 I=1,M
  210 WRITE(6,800) (D(I,J),J=1,N)
      WRITE(6,300) M,N
  300 FORMAT(1H ,2HC(,I2,1H*,I2,1H))
      DO 310 I=1,M
  310 WRITE(6,770) (C(I,J),J=1,N)
  770 FORMAT(1H ,4(12I5))
      WRITE(6,400) I
  400 FORMAT(1H ,4HA(M=,I2,1H))
      DO 410 I=1,M
  410 WRITE(6,810) A(I)
  810 FORMAT(I5)
      WRITE(6,500) J
  500 FORMAT(1H ,4HB(N=,I2,1H))
      WRITE(6,800) (B(J),J=1,N)
      CALL CATRA(M,N,A,B,L,D,C,X,U,V,W,MP,NP,IPMAX,IOPT,IZ)
      WRITE(6,5000) IOPT
 5000 FORMAT(1H0,5HIOPT=,I2)
      IF(IOPT.EQ.1) GO  TO 20
      IF(IOPT.EQ.2) GO  TO 30
      WRITE(6,6000) IPMAX
 6000 FORMAT(1H ,14HOVER ITERATION,I3)
      GO TO 50
   30 WRITE(6,7000)
 7000 FORMAT(1H ,10HINFEASIBLE)
      GO TO 50
   20 WRITE(6,8000)
 8000 FORMAT(1H0,17HOPTIMAL SOLUTION=)

      DO 40 I=1,M
   40 WRITE(6,9000) (I,J,X(I,J),J=1,N)
 9000 FORMAT(1H ,4(2HX(,I1,1H,,I1,2H)=,I3,5X))
      WRITE(6,10000) IZ
10000 FORMAT(1H0,19HOBJECTIVE FUNCTION=,I10)
   50 CONTINUE
      CALL CLOCKM(ISEC)
      WRITE(6,2700) ISEC
 2700 FORMAT(1H0,'EXEC TIME='I10)
      STOP
      END
```

6.2 デ ー タ

```
3    4
0   12   10
0   14    8
0   18    9
0   13    5
0   20    2
0    4    3
0    5    6
0   10    7
0   25    4
0   20    7
0    9    6
0    7    8
25
25
50
15
20
30
35
```

6.3 結　　果

```
M= 3
N= 4
L( 3* 4)
    0    0    0    0
    0    0    0    0
    0    0    0    0
D( 3* 4)
   12   13    5   20
   14   20   10    9
   18    4   25    7
C( 3* 4)
   10    5    6    7
    8    2    7    6
    9    3    4    8
A(M= 3)
  25
  25
  50
B(N= 4)
   15   20   30   35

IOPT= 1

OPTIMAL SOLUTION=
X(1,1)=  0     X(1,2)=  0     X(1,3)=  5     X(1,4)= 20
X(2,1)=  0     X(2,2)= 16     X(2,3)=  0     X(2,4)=  9
X(3,1)= 15     X(3,2)=  4     X(3,3)= 25     X(3,4)=  6

OBJECTIVE FUNCTION=        551
```

(注) この例題の計算時間は 24 MS であった。

6.4 数値実験の結果

Table 2 に簡単な数値実験結果を示す。

Table 2.

$M\times N$	時　間 (mS)
6×12	58
6×12	59
6×12	60
6×24	82
6×24	94
6×48	163

7. あとがき

本プログラムの作成により輸送問題を高速で解くルーチンの使用が可能になった。また本プログラムを部分ルーチンとしてより複雑な問題を解くこともできる。尚本プログラムは，著者等が開発中の応用線形計画プログラムシステム (Applied Linear Programming System, ALPS) の輸送問題コードとして使用されている。なお，使用計算機は，北大大型計算機センター Facom 230-75 である。終りに，本プログラムコードは，北大大型計算機センターライブラリ開発により，作成された。お世話を受けた北大大型計算機センターの方々に，深く謝意を表する。

文　献

1) Dantzig, G. B.: Linear Programming and Extensions (1963), Princeton University Press.
2) Hadley, G.: Linear Programming (1962), Addison-Wesley.
3) Simonnsrd, M.: Linear Programning (1966), Prentice Hall.
4) 前田治郎，大野　豊，藤井純監修：コンピュータ　マネジメント・サイエンス　ハンドブック (1971)，オーム社.