

# アーベル群と離散最適化問題

——生産可能領域の類別と  
最適解の周期的補正項——

長尾 昭 哉

## はじめに

生産・消費・投資といったような経済諸量を、連続的に変化するものとして、例えば実数によってあらわし、実数体上のベクトル空間の自然な位相や凸構造の便利な性質をうまく利用していろいろな分析を進めるのが、数理経済学の伝統である。しかしながら、固定資本財や耐久消費財などの分析において、若し分割不可能な量を exact に考慮に入れなければならない場合が生じたとすると、もはや変量の discrete な側面を無視するわけにはゆかなくなる。

現実の世界は、いわば連続性と離散性の複雑なからみ合いによって成り立っている。むしろ、経済現象を連続の相において見るか、非連続の相において見るかは、対象をとらえる側の都合によるというべきであろう。それにもかかわらず、経済学が後者にスポットを当てる機会は少かった。

このことの少くとも一つの原因は、連続の領域には analysis——微分・積分——という鋭利な武器が知られているのに反し、非連続の方面には、便利でしかも全体の見通しをよくするような分析のてだてがなかったことにあるのではなかろうか。ところが今世紀に入って、応用数学者はこの方面の構造の探究に著実な成果をつみ重ねて来た。そしてこうした諸構造の間の見通しをよくす

る上で、代数のいくつかの基礎概念の知識が有益であることも亦、一般に知られるようになった。将来、こうした知識の応用から経済学の一領域がひらかれることがないとはいえないだろう。

本稿は、整数ベクトルの或る集合が生成する群を、その部分群によって類別することが、離散型の最適化問題の解明に役立つことを、特に最適解の周期性との関連で明らかにする。(なお、以下の example の工夫と計算は本大学院社会科学研究科の鈴木牧雄君の手によるものであること、又筆者がこの主題の研究プロジェクトに着手して以来、北海道大学経済学部助手嶺野幸子氏(旧姓土倉)にいろいろお世話になったことを記して感謝の一端を表したい。)

# I

最初に、最適化問題の解を dominant part と periodic part とに分けること、後者すなわち周期的補正項をあらかじめ構成すること、などを簡単に説明しておく方がよいと思う。そこで、もっとも単純な場合つまり knapsack problem を考える。

$$\begin{aligned} \max \quad & \sum_j c_j x_j \\ \text{s.t.} \quad & \sum_j a_j x_j \leq b \\ & x_j \geq 0, \text{ integer} \end{aligned} \tag{1}$$

この問題に G. B. Dantzig 以来の伝統に従って knapsack にいろいろつめこんでハイキングに出かけるという解釈をつけてもよいし、環境資源などのコストを  $a_j$ 、財の産出などのベネフィットを  $c_j$  とするプラントがいくつかあって、コストの制約  $b$  に合せて、最も有利な組合せを選択するという問題だと解釈してもよい。

プラントのサイズをあらわす変数  $x_j$  が実数の範囲であれば、問題は実に簡単である。つまり限界生産力の最も大きいプラントを一つ選んで、そのサイズを

$b/a_i$  (ただし  $c_j/a_i = \max_j c_j/a_i$ ) とすればよいのである。

若し(1)のように  $x_j$  が整数でなければならぬとき、いいかえればプラントの単位サイズがなんらかの技術的理由であらかじめ固定されていると、解を direct に求めることは出来ず、多少とも厄介な iterative procedure——例えば dynamic programming のような——を必要とすることになる。

しかし、限界生産力最大のプラントを見出すという単純なアイディアが、ここでも実は有益なのである。何故なら若し  $b$  が十分に大きいならば、このプラントが解の dominant part を占めるであろうことは直観的に明らかだからである。

その場合、解の補正的部分が必要なことはいうまでもないが、この部分が  $b$  に関して periodic になることも亦直観的に予想される。証明は一切省略して数値例によって説明しよう。

#### example 1.

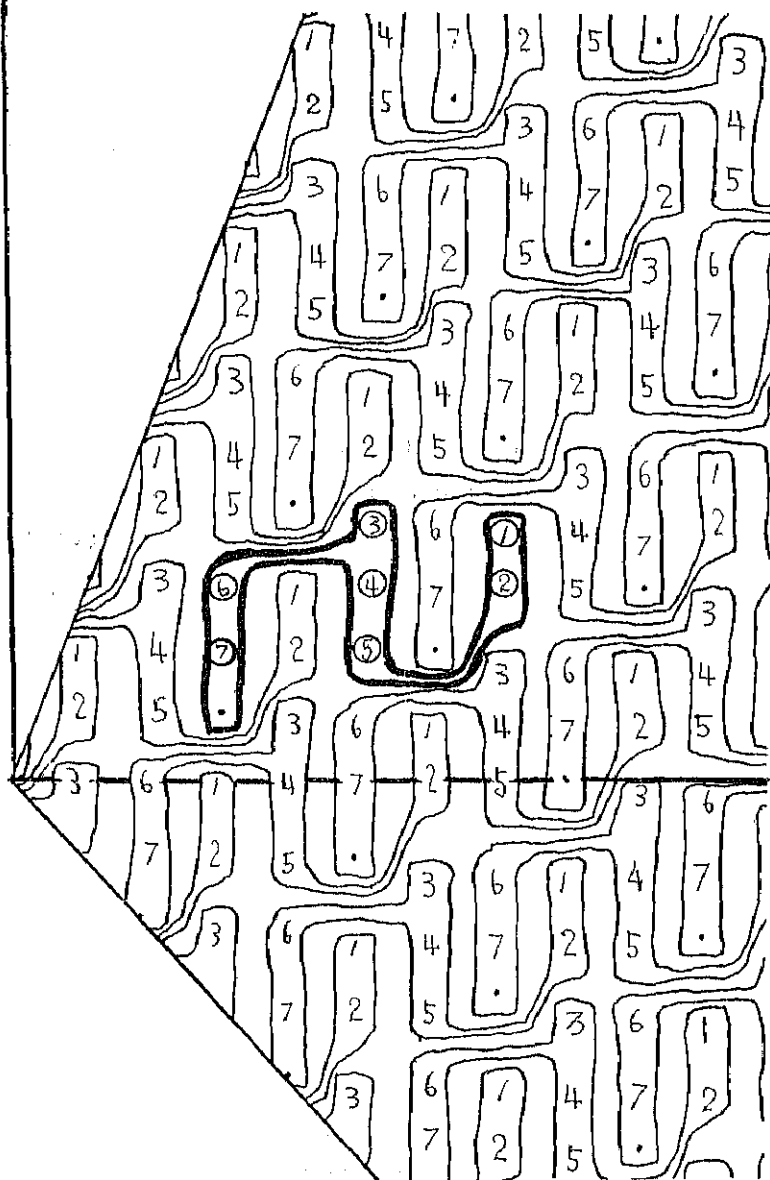
$$\begin{aligned} \max \quad z &= 9x_1 + 4x_2 + x_3 \\ 6x_1 + 3x_2 + x_3 &\leq 76 \\ x_1, x_2, x_3 &\geq 0, \text{ integer} \end{aligned}$$

第1表に上の例が解かれているが、右の部分を見ると解の内容が周期的に組み立てられていることがわかる。そしてその周期は、限界生産力最大のアクティビティがどれであるかによってきまり、この場合は  $a_1 = 6$  である。(第1表で使われているアルゴリズムについては説明を省略する——Garfinkel-Nemhauser (文献2) の第6章を見よ。)

一見してわかるように、限界生産力の最大なアクティビティ  $a_1$  をのぞく  $a_2, a_3$  の6通りの組合せが、周期的補正項をつくっている。若し例えば  $b$  が76という大きさであるなら、

$$76 \equiv 4 \pmod{6}$$





requirement space に示された生産可能領域, example 2.

## II

knapsack problem から一般の integer linear program ILP に話を進める前に、一先ず具体例によって、解の周期性が結果としてどのようにあらわれるかを visualize しておこう。

example 2.

$$\begin{aligned} \max \quad & 3x_1 + 4x_2 + 2x_3 + 4x_4 - x_5 \\ & 2x_1 + x_2 + 4x_3 + 2x_4 + x_5 = 21 \\ & -2x_1 + 3x_2 + x_4 + x_6 = 10 \\ & x_1, x_2, x_3, x_4, x_5, x_6 \geq 0, \text{ integer} \end{aligned}$$

ここで、 $x_1, \dots, x_6$  は例えば4種のプラントの設置数をあらわしているものとする。(プラントのサイズは伸縮的ではなく、従って  $x_i$  は integer であることが要求される。)

さて、example 2. の整数制約を外して linear program とし、simplex 法で解くとなると、tableau のかたちで、

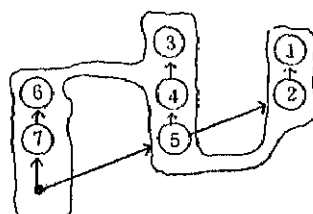
cycle 0.						
-3	-4	-2	-4	1	0	
2	1	4	2	1	0	21
-2	3	0	1	0	1	10
cycle 1.						
-17/3	0	-6/3	-8/3	1	4/3	
8/3	0	12/3	5/3	1	-1/3	53/3
-2/3	1	0	1/3	0	1/3	10/3
cycle 2.						
0	0	52/8	7/8	25/8	5/8	
1	0	12/8	5/8	3/8	-1/8	53/8
0	1	8/8	6/8	2/8	2/8	62/8

cycle 2. においてえられた最適基底とその逆行列は,

$$B = \begin{pmatrix} 2 & 1 \\ -2 & 3 \end{pmatrix} \quad B^{-1} = \begin{pmatrix} 3/8 & -1/8 \\ 2/8 & 2/8 \end{pmatrix}$$

である。

いわゆる requirement space に, LP 最適基底に含まれているベクトルの非負整数結合によって生成される LP 格子点を示してみよう。いま或る特定の requirement  $b$  が与えられて  $b$  にひとしいアクティビティの組合せを構成するとき, dominant part ないしは non-periodic part をなすのは, 第1図に・で示されているこれらの格子点である。



第 2 図

第1図でとくに注意してほしいことは, 錐体のかたちの生産可能領域が, 或るかたちのエレメントのくり返しによってカヴァされていること, いわば或る種の単調なモザイク模様になっていることである。このことの背後には, requirement space 全体が或る仕方では decompose されている事情があるのだが, そもそもこのモザイクのユニットになっているパターンはどのようにしてえられたのであろうか。

第2図は, 第1図の一つの unit pattern (太い線でかこまれた部分) を取り出している。この図は, ・によって示されている LP 格子点をいわば base

point に取って、他の七つの点にいたる path が補正ベクトルを用いて構成されるありさまを画いている。補正項の周期性とは、このようなパターンが規則的にあらわれることを指しているのである。このパターンは、最後にこの問題の network がきまったとき、その上の minimal spanning tree として画かれる。

又この図でもう一つ注意してほしいのは、すべての整数ベクトルが、 $0, 1, \dots, 7$  (ただし第1図で0点は・印であらわれている) という8種類に分類されていることである。いま各種の点の代表を取って——例えば図で太い線にかこまれている部分を pick up して——われわれの問題の最適解を具体的に書き下ろして見ると、

ラベル	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$
0	1	1	0	0	0	0
1	1	1	0	2	0	1
2	1	1	0	2	0	0
3	1	1	0	1	0	2
4	1	1	0	1	0	1
5	1	1	0	1	0	0
6	1	1	0	0	0	2
7	1	1	0	0	0	1

この解のうち、 $x_4, x_6$  によって構成される部分が periodic part ——周期的補正項——、 $x_1, x_2$  による部分が dominant part である。

以上のようなわけで、われわれの仕事は二つの段階に分れることになる。前段は、整数ベクトルの加群 module を特定の LP 最適格子に合わせて類別 decompose すること、後段は類別をおえたところで、その上に実際に解を構成することである。



以下にとりあえず問題を一般的に定式化し——ただし数学的な説明は一切附論 I にまわすことにして——更に具体例にもどって、この図に画かれた結果がえられるすじみちをたどることとする。

### III

われわれの問題——ILP——を一先ず一般的に書くと、

$$\begin{aligned} \max \quad & cx \\ Ax = & b \\ x \geq & 0, \text{ integer} \end{aligned} \quad (2)$$

問題 (2) の整数条件を外して LP として解き、最適基底を  $B$  とする。行列  $A$  は  $[B, N]$  のように分けて書くことが出来る。 $c$  も  $x$  も同様に分けるとすれば、(2) は

$$\begin{aligned} \max \quad & c_B x_B + c_N x_N \\ Bx_B + Nx_N = & b \\ x_B, x_N \geq & 0, \text{ integer} \end{aligned}$$

のように書けるが、更に

$$\begin{aligned} \max \quad & c_B B^{-1}b - (c_B B^{-1}N - c_N)x_N \\ x_B + B^{-1}Nx_N = & B^{-1}b \end{aligned}$$

と書け、ここで  $c^* = c_B B^{-1}N - c_N$  とすれば結局

$$\begin{aligned} \min \quad & c^* x_N \\ x_B = & B^{-1}b - B^{-1}Nx_N \\ x_B, x_N \geq & 0, \text{ integer} \end{aligned} \quad (3)$$

と同値になる。

ここで若し LP 最適解  $x_B = B^{-1}b$  がたまたま整数になっていれば問題は既に解かれていることになる。一般にはそういうことはないで、 $x_B \geq 0$  という条件を外して ILPC integer program over cone に変換して次へ進む。 $x_B$  にかんする非負条件を外すことは、幾何学的には或る錐体上の最適化問題に変換することを意味するのでこの名があり、又 AP asymptotic problem とも呼ばれる。つまり、 $b$  の各要素が適当に大きければ ILPC の最適解  $x_B$  は非負となり、その意味で ILPC は対応する ILP を asymptotical に解くと言えるのである。前者のどの最適解も後者の可能解とならないケースをどう扱うかについては、稿をあらためて述べたいと思う。(例えば文献 2, 252~4頁を見よ。)

$x_B \geq 0$  という条件を外したとすれば、問題は  $B^{-1}b - B^{-1}Nx_N$  が整数ベクトルであるかどうかということ、いいかえれば  $B^{-1}b$  というベクトルが  $B^{-1}Nx_N$  と modulo 1 で合同であるかどうかということになる。このことを記号を変えて、先ず次のようにあらわす。

$$\begin{aligned} \min \sum_j c_j^* x_j \\ \sum_j a_j x_j \equiv a_b \pmod{1} \end{aligned} \quad (4)$$

$B^{-1}N = [a_1, \dots, a_j, \dots, a_n]$ ,  $B^{-1}b = a_b$  である。

(4) はさらに、 $a_j = [a_j] + \bar{a}_j$  のようにわければ、 $a_j$  の整数部分  $[a_j]$  に整数  $x_j$  をかけたものは 0 と合同であるから

$$\begin{aligned} \min \sum_j c_j^* x_j \\ \sum_j \bar{a}_j x_j \equiv \bar{a}_b \pmod{1} \end{aligned}$$

$\bar{a}_j$  ( $j = 1, \dots, n$ ) の生成する群 (mod 1 の加法) を、 $\{\bar{a}\}$  とすれば、この問題は結局  $\{\bar{a}\}$  の元の非負整数結合によって元  $\bar{a}_b$  をつくるという問題になる。群  $\{\bar{a}\}$  は  $\{B\}$  による  $A$  の類別  $\{A\}/\{B\}$  にひとしいことを附論 IA で述べる。

## IV

先に述べたように、われわれの仕事の前半部は群  $G$  を確定し、後半部つまりその群の各要素について最適化問題を解くという段階に進むことが出来るようにすることである。 $G$  は kernel  $f$  による  $\{A\}$  の decomposition によってつくられるのだが、実はその同じ  $G$  をもっとわかりやすくつくる方法がある。今までの方法を第1の方法と呼び、これを第2の方法と呼ぶことにしよう。

いま  $S = \{x | Bx = b, x \text{ integer}\}$ ,  $T = \{y | \bar{B}y = \bar{b}, y \text{ integer}\}$  という2つの集合があるとする。ここで  $B$  および  $\bar{B}$  は  $m$  次元の整数行列。 $b$  および  $\bar{b}$  は同次元の整数ベクトルである。 $S$  と  $T$  それぞれの要素の一対一対応が  $y = Px$  ( $P$  は  $m$  次元整数行列, unimodular) で与えられるとき  $S$  と  $T$  は同値な表現という。(文献 1.259頁, 定理1)

われわれは、あたえられた integer program のさまざまな equivalent representation のうち或る都合のよい表現をえたいわけであるが、そのためには、同値性を保ちながら次々と表現をかえてゆく操作を知っていなければならない。それが附論Ⅱで定義される基本操作—— row and column elementary operation ——である。以下の数値例で見るとわかりやすいのだが、行の基本操作は次に述べる行列  $R$  を左から掛けること、又列の基本操作は  $C$  を右から掛けることに相当する。

基本操作定理:  $R$  と  $C$  を  $m$  次元整数行列で unimodular であるとし,  $RBC = \bar{B}$  であるとする。そうすると,

$$S = \{x | Bx = b, x \text{ integer}\}$$

$$T = \{y | \bar{B}y = Rb, y \text{ integer}\}$$

は同値な表現であり、 $S$ と $T$ それぞれの要素の一対一対応は  $y=C^{-1}x$  によってあたえられる。

さて、われわれは行又は列についての基本操作を次々に加えながら、いま述べた都合のよい表現、いわゆる canonical representation を求める。具体的にいうと、附論Ⅱで述べる手続きに従って $B$ を Smith normal form に変換する。

スミスの標準型とは、次のような正方行列をいう。

(1) 非対角要素はすべて 0,

(2) 対角要素  $\delta_1, \delta_2, \dots, \delta_m$  は正の整数で  $\delta_{i+1} \equiv 0 \pmod{\delta_i}$ , すなわち  $\delta_{i+1}$  は  $\delta_i$  で割り切れる。

$B$ がこのような標準型に変わったとき、(4)の制約式は同値な次の表現に変わる。

$$\bar{B}y = R(b - Nx_N), \quad y \text{ integer} \quad (5)$$

ただし  $\bar{B} = RBC$ ,  $y = C^{-1}x_B$  である。 $R$ の第  $i$  行を  $R_i$  とすれば、(5)の第  $i$  行は

$$\delta_i y_i = R_i(b - Nx_N)$$

これをさらに書きあらためると

$$R_i Nx_N \equiv R_i b \pmod{\delta_i} \quad i = 1, \dots, m$$

そしてここで若し  $\delta_i = 1$ ,  $i = 1, \dots, k-1$  であると  $i = 1, \dots, k-1$  の制約式は必ず満たされるから不要である。従って、

$$\alpha_{ij} \equiv R_i N_j \pmod{\delta_i}$$

$$\alpha_{i0} \equiv R_i b \pmod{\delta_i}$$

とすれば (4) は結局

$$\min \sum c_j x_j,$$

$$\sum_{j=1}^n \alpha_{ij} x_j \equiv \alpha_{i0} \pmod{\delta_i} \quad i = k, \dots, m$$

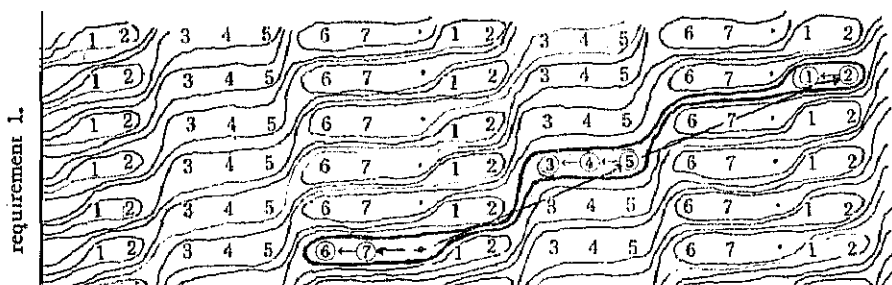
或いは以下の数値例の書き方に従えば,

$$\begin{pmatrix} \alpha_{k1} \\ \vdots \\ \alpha_{m1} \end{pmatrix} + \dots + \begin{pmatrix} \alpha_{kn} \\ \vdots \\ \alpha_{mn} \end{pmatrix} \equiv \begin{pmatrix} \alpha_{k0} \\ \vdots \\ \alpha_{m0} \end{pmatrix} \pmod{\begin{pmatrix} \delta_k \\ \vdots \\ \delta_m \end{pmatrix}}$$

のようになる。

われわれはこれで仕事の前半を終える。すなわち LP 最適基底  $B$  の Smith normal form への変形によって群  $\langle Z \rangle / \langle B \rangle$  のいわゆる canonical representation  $G(\delta_k, \dots, \delta_m)$  を見出したわけである。

後半の仕事にうつる前に、もとの問題の非基底ベクトルと  $G$  の元との対応をつける。すなわち  $G(\delta_k, \dots, \delta_m)$  の元を



canonical representation, example 2.

requirement 2.

$$g(i_k, \dots, i_m)_i \\ 0 < i_k \leq \delta_k - 1, 0 < i$$

のようにあらわせば,

$$\alpha_j \rightarrow g(\alpha_{kj}, \dots, \alpha_{mj})$$

ただし  $\alpha_j$  は行操作  $R$  によって変換されたものの非基底ベクトル  $N_j$  である。  
ここで若し 2 つ以上のベクトルが同一の  $G$  の元に対応することがあれば,  $c_j^*$  を考慮に入れて一つをえらべばよい。以上の第 2 の方法を, 実際に数値例で具体的に説明する。

exampl 2. の制約条件は, 新しい tableau form で次のように書かれる。

この form の内容は,

$$[B|N|b], [R], [C]$$

である。(cycle から cycle への演算の規則は附論 II を見よ。)

$$\text{cycle 0.} \\ \left[ \begin{array}{cc|cc} 2 & 1 & 4 & 2 \\ -2 & 3 & 0 & 1 \end{array} \right] \begin{array}{cc} 21 \\ 10 \end{array} \left[ \begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array} \right] \left[ \begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array} \right]$$

第 1 列と第 2 列を入れ替える。  $C_1 \rightleftharpoons C_2$  ( $\{B\}$  の基底の入れ替え操作)

$$\text{cycle 1.} \\ \left[ \begin{array}{cc|cc} 1 & 2 & 4 & 2 \\ 3 & -2 & 0 & 1 \end{array} \right] \begin{array}{cc} 21 \\ 10 \end{array} \left[ \begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array} \right] \left[ \begin{array}{cc} 0 & 1 \\ 1 & 0 \end{array} \right]$$

第 1 行の 3 倍を第 2 行から引く。  $R_2 \leftarrow 3R_1 \rightarrow R_1$  ( $\{Z\}$  の基底の変換)

$$\text{cycle 2.} \\ \left[ \begin{array}{cc|cc} 1 & 2 & 4 & 2 \\ 0 & -8 & -12 & -5 \end{array} \right] \begin{array}{cc} 21 \\ -53 \end{array} \left[ \begin{array}{cc} -1 & 0 \\ -3 & 1 \end{array} \right] \left[ \begin{array}{cc} 0 & 1 \\ 1 & 0 \end{array} \right]$$

第 1 列の 2 倍を第 2 列から引く。  $C_2 \leftarrow 2C_1 \rightarrow C_2$  ( $\{B\}$  の基底の変換)

$$\text{cycle 3.} \\ \left[ \begin{array}{cc|cc} 1 & 0 & 4 & 2 \\ 0 & -8 & -12 & -5 \end{array} \right] \begin{array}{cc} 21 \\ -53 \end{array} \left[ \begin{array}{cc} -1 & 0 \\ -3 & 1 \end{array} \right] \left[ \begin{array}{cc} 0 & 1 \\ 1 & -2 \end{array} \right]$$

第 2 行の符号を変える。  $-R_2 \rightarrow R_2$  ( $\{Z\}$  の基底の変換)

cycle 4.

$$\left[ \begin{array}{ccc|ccc} 1 & 0 & 4 & 2 & 1 & 0 \\ 0 & 8 & 12 & 5 & 3 & -1 \end{array} \right] \left[ \begin{array}{cc} 1 & 0 \\ 3 & -1 \end{array} \right] \left[ \begin{array}{cc} 0 & 1 \\ 1 & -2 \end{array} \right]$$

$B$ の部分が normal form になったので,  $G$ の canonical な表現  $G(8)$  が明らかになった。制約式を (6) のかたにすると,

$$4x_3 + 5x_4 + 3x_5 + 7x_6 \equiv 5 \pmod{8}$$

従って非基底ベクトル  $\alpha_j$  と module の元  $g(i_j)$  のとの対応は,

$$\alpha_3 \longrightarrow g(4)$$

$$\alpha_4 \longrightarrow g(5)$$

$$\alpha_5 \longrightarrow g(3)$$

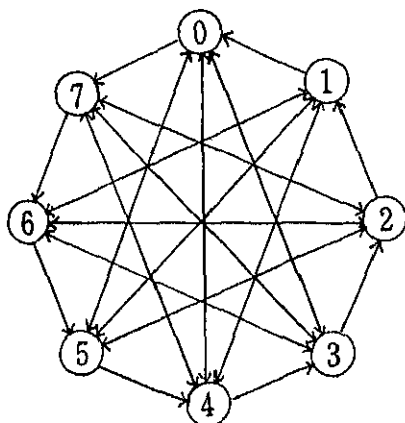
$$\alpha_6 \longrightarrow g(7)$$

こうして,  $i = 0, \dots, 7$  について  $g(i) \rightarrow g(i) + g(4)$ ,  $g(i) \rightarrow g(i) + g(5)$ ,  $g(i) \rightarrow g(i) + g(3)$ ,  $g(i) \rightarrow g(i) + g(7)$  という edge がつくられ,  $G$ の上の問題をネットワーク  $N(G, \alpha)$  の上の問題にうつすことが出来る。 $G$ の上の最小化問題は dynamic programming で処理した方が, 実用上の利点が多いが, 本稿はむしろさまざまな数学的構造間のつながりを説明することがねらいなので, network 上の shortest route problem というかたちで扱う。

## V

仕事の後半部つまり最適解を一つ一つ求める部分は, ネットワーク  $N(G, \alpha)$  をつくることから始まる。すなわち  $G$ の元—— example 2では  $g(0), \dots, g(7)$  ——を node とし, 問題の非基底ベクトル——  $\alpha_3, \alpha_4, \alpha_5, \alpha_6$  ——を edge とするグラフである。(第3図)

第4図A, ..., Hは, Dijkstra のアルゴリズムによって各 node への最短路



第 3 図

がきまり, minimal tree が張られてゆくプロセスを示す。(アルゴリズムの要点は附論Ⅲに記すがくわしくは例えば文献(1, 3)などを参照されたい。第2表は図に示した計算の内容である。)

第 2 表

step, tree nodes, neighbors: distance

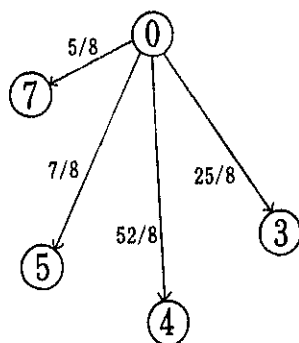
0.	$g(0)$	$g(3) : 25/8$ $g(4) : 52/8$ $g(5) : 7/8$ $g(7) : 5/8 *$
1.	$g(0)$ $g(7)$	$g(3) : \min (25/8, 57/8) = 25/8$ $g(4) : \min (52/8, 12/8) = 12/8$ $g(5) : 7/8 *$ $g(2) : 30/8$ $g(6) : 10/8$



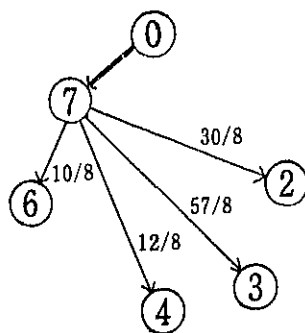
2.	$g(0)$	$g(3) : 25/8$
	$g(7)$	$g(4) : \min (12/8, 12/8) = 12/8$
	$g(5)$	$g(2) : \min (30/8, 14/8) = 14/8$
		$g(6) : 10/8 \quad *$
		$g(1) : 59/8$
3.	$g(0)$	$g(3) : \min (25/8, 17/8) = 17/8$
	$g(7)$	$g(4) : 12/8 \quad *$
	$g(5)$	$g(2) : \min (14/8, 62/8) = 14/8$
	$g(6)$	$g(1) : \min (59/8, 35/8) = 35/8$
4.	$g(0)$	$g(3) : \min (17/8, 17/8) = 17/8$
	$g(7)$	$g(2) : 14/8 \quad *$
	$g(5)$	$g(1) : \min (35/8, 19/8) = 19/8$
	$g(6)$	
	$g(4)$	
5.	$g(0)$	$g(3) : 17/8 \quad *$
	$g(7)$	$g(1) : \min (19/8, 19/8) = 19/8$
	$g(5)$	
	$g(6)$	
	$g(4)$	
	$g(2)$	
6. 7	省略	

tie-breaking rule: 後からつくられた neighbor への route がより小さいとみなす。

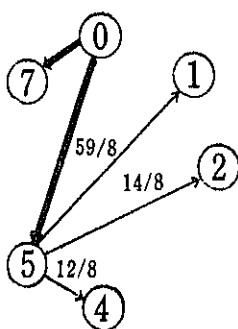
$N(G, \alpha)$  の上に画かれた tree がもとの整数ベクトル空間 (canonical) にもどされたのが第 5 図であり, 更に基底変換以前の空間に復原したものが冒頭の第 1 図である。はじめに示された周期的補正項のモザイク模様は, 要するに以上の道程をへて到達した最後の結果なのである。最後に  $G$  そのものが巡回群にならないような example をひとつつけ加えて稿を終える。



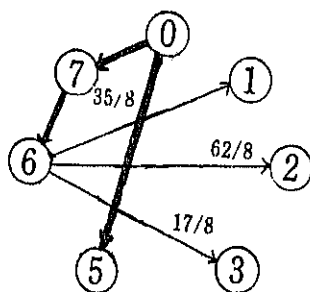
step 0



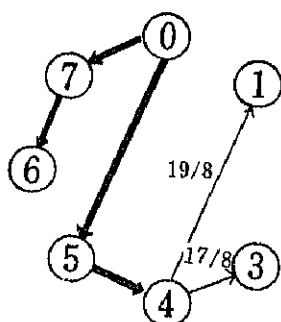
step 1



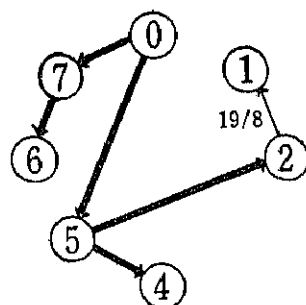
step 2



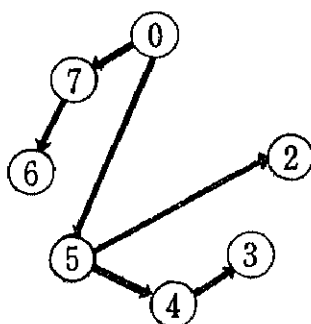
step 3



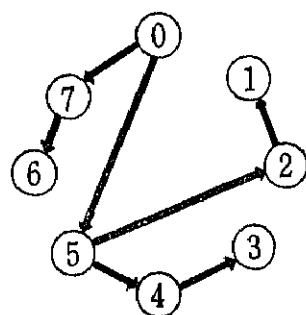
step 4



step 5



step 6



step 7

example 3.

$$\begin{aligned}
 \max \quad & 3x_1 + 4x_2 + 2x_3 + 2x_4 \\
 & 4x_1 + 2x_2 + 2x_3 + x_4 + x_5 = 15 \\
 & 2x_1 + 4x_2 + 3x_3 + 6x_4 + x_6 = 24 \\
 & x_1, x_2, x_3, x_4, x_5, x_6 \geq 0, \text{ integer}
 \end{aligned}$$

線型計画問題とみなして解く。

cycle 0.

-3	-4	-2	-2	0	0	
4	2	2	1	1	0	15
2	4	3	6	0	1	24

cycle 1.

-1	0	1	4	0	1	
3	0	1/2	-2	1	-1/2	3
2/4	1	3/4	6/4	0	1/4	6

cycle 2.

0	0	7/6	20/6	2/6	5/6	
1	0	1/6	-4/6	2/6	-1/6	1
0	1	4/6	11/6	-1/6	2/6	11/2

$$B = \begin{pmatrix} 4 & 2 \\ 2 & 4 \end{pmatrix} \quad B^{-1} = \begin{pmatrix} 2/6 & -1/6 \\ -1/6 & 2/6 \end{pmatrix}$$

$B$ がわかったので それをスミスの標準型に変換する。計算のチェックのために  $R$  と  $C$  を右辺に追加する。

cycle 0.

$$\left[ \begin{array}{cc|cc|cc|c} 4 & 2 & 2 & 1 & 1 & 0 & 15 \\ 2 & 4 & 3 & 6 & 0 & 1 & 24 \end{array} \right] \left[ \begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array} \right] \left[ \begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array} \right]$$

第 1 列と第 2 列を入れ替える。 $C_1 \rightleftharpoons C_2$  ( $\{B\}$  の基底の入れ替え操作)

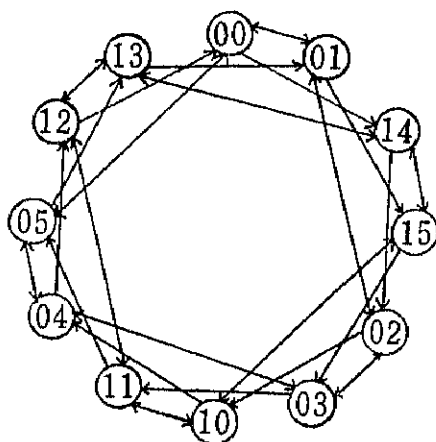
cycle 1.

$$\left[ \begin{array}{cc|cc|cc|c} 2 & 4 & 2 & 1 & 1 & 0 & 15 \\ 4 & 2 & 3 & 6 & 0 & 1 & 24 \end{array} \right] \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

第 1 列の 2 倍を第 2 列から引く。 $C_2 - 2C_1 \rightarrow C_2$

cycle 2.

$$\left[ \begin{array}{cc|cc|cc|c} 2 & 0 & 2 & 1 & 1 & 0 & 15 \\ 4 & -6 & 3 & 6 & 0 & 1 & 24 \end{array} \right] \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & -2 \end{bmatrix}$$



第 6 図

第 1 行の 2 倍を第 2 行から引く。 $R_2 - 2R_1 \rightarrow R_2$  ( $\{Z\}$  の基底ベクトルの変換)

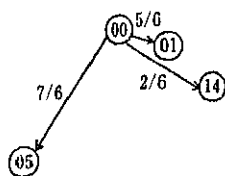
cycle 3.

$$\left[ \begin{array}{cc|cc|cc|c} 2 & 0 & 2 & 1 & 1 & 0 & 15 \\ 0 & -6 & -1 & 4 & -2 & 1 & -6 \end{array} \right] \begin{bmatrix} 1 & 0 \\ -2 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & -2 \end{bmatrix}$$

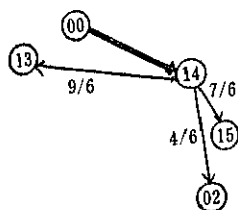
第 2 列を  $-1$  倍する。 $-C_2 \rightarrow C_2$

cycle 4.

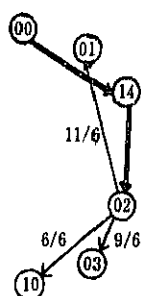
$$\left[ \begin{array}{cc|cc|cc|c} 2 & 0 & 2 & 1 & 1 & 0 & 15 \\ 0 & 6 & -1 & 4 & -2 & 1 & -6 \end{array} \right] \begin{bmatrix} 1 & 0 \\ -2 & 1 \end{bmatrix} \begin{bmatrix} 0 & -1 \\ 1 & 2 \end{bmatrix}$$



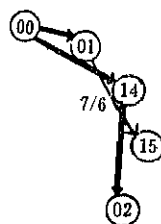
step 0



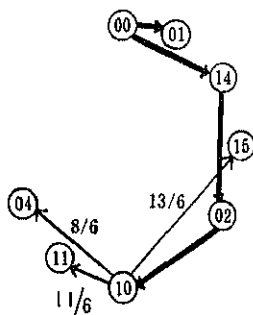
step 1



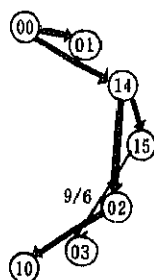
step 2



step 3

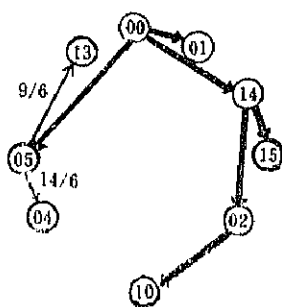


step 4

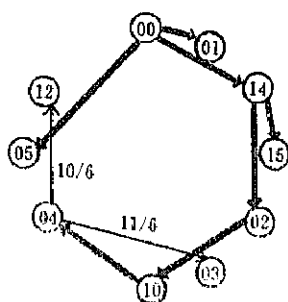


step 5

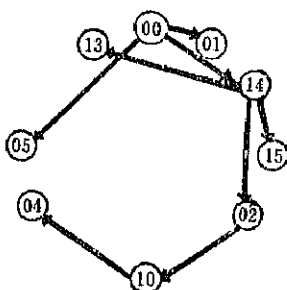
第 7 图 ——分位



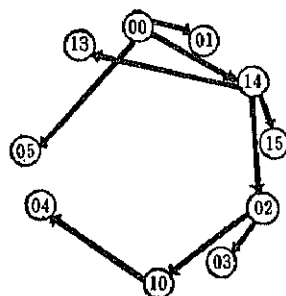
step 6



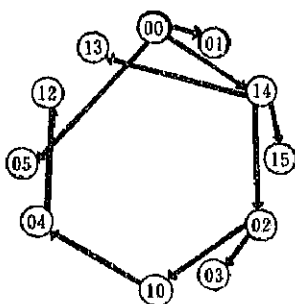
step 7



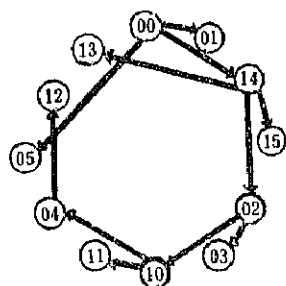
step 8



step 9



step 10



step 11

$B$  の normal form がえられ,  $G$  の canonical representation  $G(2, 6)$  がきまる。  
 制約式を書きあらためる。

$$\begin{pmatrix} 0 \\ 5 \end{pmatrix} x_3 + \begin{pmatrix} 1 \\ 4 \end{pmatrix} x_4 + \begin{pmatrix} 1 \\ 4 \end{pmatrix} x_5 + \begin{pmatrix} 0 \\ 1 \end{pmatrix} x_6 \equiv \begin{pmatrix} 1 \\ 0 \end{pmatrix} \pmod{\begin{pmatrix} 2 \\ 6 \end{pmatrix}}$$

$\alpha_j$  について計算をチェックする。

$$\alpha_3: \begin{pmatrix} 1 & 0 \\ -2 & 1 \end{pmatrix} \begin{pmatrix} 2 \\ 3 \end{pmatrix} = \begin{pmatrix} 2 \\ -1 \end{pmatrix} \equiv \begin{pmatrix} 0 \\ 5 \end{pmatrix} \pmod{\begin{pmatrix} 2 \\ 6 \end{pmatrix}}$$

$$\alpha_4: \begin{pmatrix} 1 & 0 \\ -2 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 6 \end{pmatrix} = \begin{pmatrix} 1 \\ 4 \end{pmatrix} \equiv \begin{pmatrix} 1 \\ 4 \end{pmatrix} \pmod{\begin{pmatrix} 2 \\ 6 \end{pmatrix}}$$

$$\alpha_5: \begin{pmatrix} 1 & 0 \\ -2 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ -2 \end{pmatrix} \equiv \begin{pmatrix} 1 \\ 4 \end{pmatrix} \pmod{\begin{pmatrix} 2 \\ 6 \end{pmatrix}}$$

$$\alpha_6: \begin{pmatrix} 1 & 0 \\ -2 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \equiv \begin{pmatrix} 0 \\ 1 \end{pmatrix} \pmod{\begin{pmatrix} 2 \\ 6 \end{pmatrix}}$$

$$\alpha_0: \begin{pmatrix} 1 & 0 \\ -2 & 1 \end{pmatrix} \begin{pmatrix} 15 \\ 24 \end{pmatrix} = \begin{pmatrix} 15 \\ -6 \end{pmatrix} \equiv \begin{pmatrix} 1 \\ 0 \end{pmatrix} \pmod{\begin{pmatrix} 2 \\ 6 \end{pmatrix}}$$

こうして  $\alpha_j$  と  $g(i_1, i_2)$  との対応がきまる。

$$\alpha_3 \longrightarrow g(0\ 5)$$

$$\alpha_4, \alpha_5 \longrightarrow g(1\ 4)$$

$$\alpha_6 \longrightarrow g(0\ 1)$$

$$\alpha_0 \longrightarrow g(1\ 0)$$

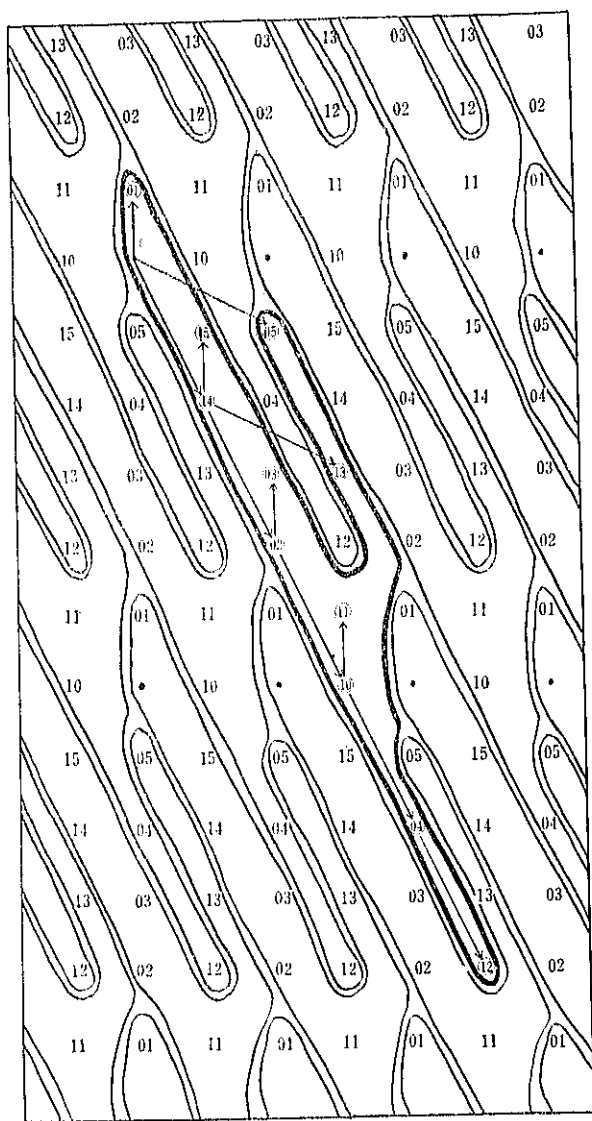
$N(G, \alpha)$  は第 6 図のよう画かれ, その上で  $G$  の各元への minimal tree が第 7 図の step 0, ..., 11 の順に張られる。(附論 III (第 3 表) にこの計算の内容を記す。)

第 7 図 step 11 の minimal spanning tree が整数空間にもとざれると第 8 図のようになり, 更に基底変更のまえのかたちで示せば第 9 図のパターンになる。(ほんらいの生産可能領域の decomposition) example 3 の周期的補正項は,

$\alpha_0$ に対応する $G$ の元: $(x_3, \quad x_4, \quad x_5, \quad x_6)$				
$g(00):$	( 0	0	0	0 )
$g(01):$	( 0	0	0	1 )



$g(02) :$	( 0	0	2	0 )
$g(03) :$	( 0	0	2	1 )
$g(04) :$	( 0	0	4	0 )
$g(05) :$	( 1	0	0	0 )
$g(10) :$	( 0	0	3	0 )
$g(11) :$	( 0	0	3	1 )
$g(12) :$	( 0	0	5	0 )
$g(13) :$	( 1	0	1	0 )
$g(14) :$	( 0	0	1	0 )
$g(15) :$	( 0	0	1	1 )



第 8 图

## 附 論 I

本論の主要部分にあらわれる線型代数学上の議論のあらすじを述べる。(ただし概念の定義や定理の証明については、それぞれ特色ある簡略化の工夫の上で、Hu, Garfinkel-Nemhauser, Salkin, 土倉, (文献1~4)にまとめられているし、より厳密な説明は van der Waerder. の linear algebra にかんする章にある。) IA は、或る integer program に特有な群  $G$  が、整数ベクトルの集合が生成する群を、そのある部分群によって類別することから確定される仕方であって、これはいわゆる準同型定理の応用である。IB は上述の群  $G$  が巡回群の直和に分解されるという話であって、これはアーベル群の基本定理と呼ばれるものの一つ具体化である。

### I A

経済学者になじみ深い実数ベクトル空間  $R^n$  がはじめに前提されていることにしよう。最初の出発点は、或る  $m$  次元ベクトルの集合がつねにベクトルの加法において加(法)群 module, abelian group を生成するということである。

行列  $A$  の列ベクトルの集合  $\{A_1, A_2, \dots\}$  を便宜上同じ記号  $A$  で示すことにし、その集合  $A$  の generate する module を  $\{A\}$  であらわすことにする。いかえれば、集合  $\{Ax \mid x \in Z\}$  はベクトルの加法に関して module ——  $Z$  を作用域に持つ加群——をなす。なおこの文中、 $Z$  は整数の集合、又は文脈によってきまる適当な次元を持つ整数ベクトル全体の集合をあらわすことにする。 $\{A\}$  は一般には  $\{Z\}$  の部分群になるが、example 2. のように  $A$  が単位行列を含む場合は  $\{Z\}$  と一致する。

LP 最適基底  $B$  がきまると上と同様の仕方で集合  $B$  は module  $\{B\}$  を生成

する。 $\{B\}$  は  $\{A\}$  の正規部分群であるから剰余群  $\{A\}/\{B\}$  を定めることができる。これがわれわれの必要とする  $G$  である。

行列  $B^{-1}A$  の列ベクトルの集合を  $a = \{a_1, a_2, \dots, a_n\}$  であらわし、さらにそれらの小数部分の集合  $\bar{a} = \{\bar{a}_j \mid \bar{a}_j = a_j - [a_j]\}$  が modulo 1 の加法によって生成する群  $\{\bar{a}\}$  を考える。変換  $B^{-1}$  と fractional part を取る変換  $\phi$  との合成に  $f = \phi B^{-1}$  よっては  $\{A\}$  は  $\{\bar{a}\}$  に写される。この  $f$  は準同型写像 homomorphism になる。そうするとよく知られた次の準同型定理によって、

$$\frac{\{A\}}{\text{kernel } f} = \frac{\{A\}}{\{B\}} \cong G$$

つまりこうして、第1の方法によってえられる群がどのようなものであるかがわかるわけである。

準同型定理:  $f: G \rightarrow G'$  を homomorphism とすれば、

$$G/\text{kernel } f \cong \text{image } f.$$

ここで image  $f$  は  $G$  の元の像の全体で  $G'$  の部分群となる。もしこれが  $G'$  と一致するとき  $G'$  は  $G$  の或る剰余群と同型である。

## I B

$R^n$  の単位ベクトルの集合  $E = \{e_1, \dots, e_m\}$  は module  $\{Z\}$  の basis のひとつと考えることができる。 $\{Z\}$  の各元を  $E$  の一次結合であらわす方法が unique にきまるという意味であるから、このことは  $\{Z\}$  が直和 direct sum に分解されることと同値である。すなわち

$$\{Z\} = e_1 Z \oplus \dots \oplus e_m Z$$

ここで  $e_j Z$  はベクトル  $e_j$  と集合  $Z$  との積すなわち  $e_j$  によって generate される cyclic group  $\langle e_j \rangle$  をあらわす。

さてⅣに述べた  $B$  の Smith normal form への変形によって  $B$  を  $\bar{B} = RBC$  に変えることは,  $R$  によって  $\{Z\}$  と  $\{B\}$  の基底を変えることである。すなわち  $RE = E'$  とすれば

$$\{Z\} = e_1' Z \oplus \cdots \oplus e_m' Z$$

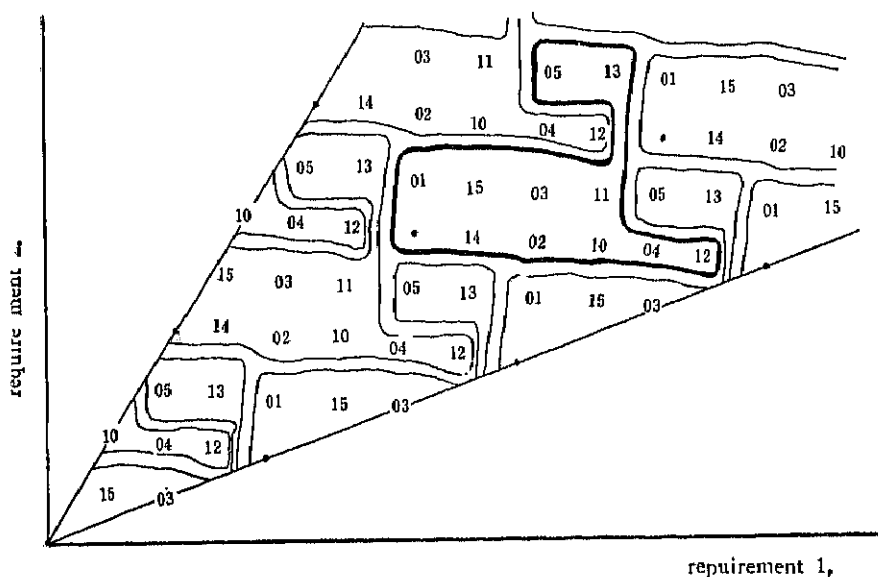
となり, また  $\{B\} \cong \{\bar{B}\}$  であるから

$$\begin{aligned} \{B\} &\cong \bar{b}_1 Z \oplus \cdots \oplus \bar{b}_m Z \\ &= \delta_1 e_1' Z \oplus \cdots \oplus \delta_m e_m' Z \\ &= e_1' (\delta_1 Z) \oplus \cdots \oplus e_m' (\delta_m Z) \end{aligned}$$

である。巡回群  $e_i' (\delta_i Z)$  は巡回群  $e_i' Z$  の部分群であることはいうまでもない。

従って  $\{Z\} / \{B\}$  が次のような直和に分解されることになる。

$$\frac{\{Z\}}{\{B\}} \cong \frac{e_1' Z}{e_1' (\delta_1 Z)} \oplus \cdots \oplus \frac{e_m' Z}{e_m' (\delta_m Z)}$$



生産可能領域, example 3.

第 9 図

$$\cong \frac{Z}{\delta_1 Z} \oplus \cdots \oplus \frac{Z}{\delta_m Z}$$

$$\cong G(\delta_1) \oplus \cdots \oplus G(\delta_m)$$

このような分解はより一般的なかたちで、いわゆるアーベル群の基本定理 the Fundamental Theorem of Abelian Groups with Finitely Many Generators に一般化することが出来る。(文献5, 8頁)

## 附 論 II

整数正方向行列  $B$  は、次のような基本操作をくり返すことによってスミスの標準型に変形される。

- (1) ある行(列)を他の行(列)と入れ替える。
- (2) ある行(列)の整数倍を他の行(列)に加える。
- (3) ある行(列)の符号を変える。

変形のアルゴリズムは、最大公約数のためのいわゆるユークリッドの互除法 Euclidean Algorithm と本質的に同じもので、 $B$  を対角化を終った  $B_{11}$  と、これから対角化さるべき  $B_{22}$  とに分けて、

$$B = \left[ \begin{array}{c|c} B_{11} & 0 \\ \hline 0 & B_{22} \end{array} \right]$$

としながら、 $B_{22}$  について以下の step 1~6 をくりかえし適用するというものである。

- step 1. 基本操作(1)により、 $b_{11} = \min_{i,j} |b_{ij}|$  となるよう入れ替える。 $b_{11} < 0$  の時、基本操作(3)により正にする。step 2へ。
- step 2.  $b_{i1} = n_{i1} b_{11}$  ( $i=2, \dots, n$ ;  $n_{i1}$  は整数)、そして、 $b_{1j} = n_{1j} b_{11}$  ( $j=2, \dots, n$ ;  $n_{1j}$  は整数)である時、
- $$r_i = r_i - n_{i1} r_1 \quad (i=2, \dots, n)$$
- $$c_j = c_j - n_{1j} c_1 \quad (j=2, \dots, n)$$

とし (基本操作 (2)), step 4 へ進む。そうでない時は, step-3 へ進む。

step 3.  $b_{ij} \Leftarrow n_{ij} b_{11}$  (あるいは,  $b_{ij} \Leftarrow n_{ij} b_{11}$ ) である任意の  $i(j)$  を選び,

$$r_i = r_i - n_{ij} r_1 < r_1$$

$$(c_j = c_j - n_{ij} c_1 < c_1)$$

とし (基本操作 (2)), step 1 にもどる。

step 4.  $b_{11} = \min_{i,j} |b_{ij}|$  である時は, step 5 へ進む。そうでない時は, step 1 へもどる。

step 5.  $b_{ij} = n_{ij} b_{11}$  ( $i, j = 2, \dots, n; n_{ij}$  は整数) である時, 終了する。ある  $i, j$  について,  $b_{ij} \Leftarrow n_{ij} b_{11}$  の時は, step 6 へ進む。

step 6.  $b_{ij} \Leftarrow n_{ij} b_{11}$  である任意の  $i, j$  を選ぶ。そして,

$$r_i = r_i + n_{ij} r_1 \quad (0 < b_{ij} - n_{ij} b_{11} < b_{11})$$

$$c_j = c_j - c_1$$

とし (基本操作 (2)), step 1 へもどる。

example

$$\begin{bmatrix} -6 & -3 & 11 \\ 4 & 1 & 3 \\ 11 & 2 & 18 \end{bmatrix}$$

step 1: 第 1 行と第 2 行, 第 1 列と第 2 列を入れ替える。  $r_1 \leftrightarrow r_2, c_1 \leftrightarrow c_2$

$$\begin{bmatrix} 1 & 4 & 3 \\ -3 & -6 & 11 \\ 2 & 11 & 18 \end{bmatrix}$$

step 2: 第 1 行の 3 倍を第 2 行に加え, 第 1 行の 2 倍を第 3 行から引く。  $r_2 + 3r_1$   
 $\rightarrow r_2, r_3 - 2r_1 \rightarrow r_3$

$$\begin{bmatrix} 1 & 4 & 3 \\ 0 & 6 & 20 \\ 0 & 3 & 12 \end{bmatrix}$$

step 2 (つづき) : 第 1 列の 4 倍を第 2 列から引き, 第 1 列の 3 倍を第 3 列から引  
 $c_2 - 4c_1 \rightarrow c_2, c_3 - 3c_1 \rightarrow c_3$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 6 & 20 \\ 0 & 3 & 12 \end{bmatrix}$$

step 1: 第2行と第3行を入れ替える。 $r_2 \rightleftharpoons r_3$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 3 & 12 \\ 0 & 6 & 20 \end{bmatrix}$$

step 2: 第2行の2倍を第3行から引く。 $r_3 - 2r_2 \rightarrow r_3$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 3 & 12 \\ 0 & 0 & -4 \end{bmatrix}$$

step 2 (つづき): 第2列の4倍を第3列から引く。 $c_3 - 4c_2 \rightarrow c_3$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & -4 \end{bmatrix}$$

step 4: そのまま。

step 6: 第2行の2倍を第3行から引く。 $r_3 - 2r_2 \rightarrow r_3$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & -6 & -4 \end{bmatrix}$$

step 6 (つづき): 第3列から第2列を引く。 $c_3 - c_2 \rightarrow c_3$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 3 & -3 \\ 0 & -6 & 2 \end{bmatrix}$$

step 1: 第2行と第3行, 第2列と第3列を入れ替える。 $r_2 \rightleftharpoons r_3, c_2 \rightleftharpoons c_3$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & -6 \\ 0 & -3 & 3 \end{bmatrix}$$

step 3: 第2行の2倍を第3行に加える。 $r_3 + 2r_2 \rightarrow r_3$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & -6 \\ 0 & 1 & -9 \end{bmatrix}$$

step 1: 第2行と第3行を入れ替える。 $r_2 \rightleftharpoons r_3$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -9 \\ 0 & 2 & -6 \end{bmatrix}$$

step 2: 第2行の2倍を第3行から引く。 $r_3 - 2r_2 \rightarrow r_3$



$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -9 \\ 0 & 0 & 12 \end{bmatrix}$$

step 2 (つづき) : 第 2 列の 9 倍を第 3 列に加える。  $c_3 + 9c_2 \rightarrow c_3$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 12 \end{bmatrix}$$

step 4: そのまま。

step 5: そのまま。——終了。

### 附 論 III

本稿にいうネットワーク  $N(G, \alpha)$  とは,  $|G|$  個の点 node の集合と  $\alpha_j$  の定める方向付けられた辺 directed edge の集合から成る configuration である。 $G$  の単位元に対応する node が source となり, sink は  $\alpha_0$  によって定められる場合もある。

edge は非基底ベクトル  $\alpha_j$  に対応する  $G$  の元  $G(\alpha_{k_1}, \dots, \alpha_{m_j})$  による演算のすべてによって定まる。以下簡単のために  $g$  を単純な suffix によって  $g_i, g_j, \dots$  のように特定するとすれば,  $\alpha_j \rightarrow g_j$  のとき

$$g_i + g_j = g_k \quad (G \text{ の演算})$$

ならば,  $g_i \rightarrow g_k$  という辺がつくということになる。それぞれの辺には  $C_{i*}$  という距離がつけられる。若し二つ以上の非基底ベクトルと一つの  $g_j$  の対応があるならば, それらのベクトルのうち  $C_{i*}$  の最少のもの一つをえらべばよい。

以下の説明において,  $g_i$  から  $g_j$  への辺があるとき  $g_j$  を  $g_i$  の neighbor という。又,  $g_i$  から  $g_j$  への chain とは,  $g_i$  から  $g_j$  にいたる node と edge の集合で  $g_i$  からは一つの出てゆく辺,  $g_j$  には一つの入ってくる辺, その他の点には入る辺と出る辺が一つずつあるものをいう。又,  $g_i$  から  $g_j$  への chain のうち, 距離が最小のものを minimal chain と呼び, そのような chain を

つくっている辺と点のすべてを一しにして minimal spanning tree という。

アルゴリズムは source から始めて最も近い node を見つけ、次に 2 番目に近い node を見つける、というように進む。いま  $g_i$  を  $g_0$  にもっとも近い点とすると先ず minimal chain  $(g_0, g_i)$  が出来る。次に、 $g_0$  又は  $g_i$  の neighbor のうちの或る点  $g_j$  が 2 番目に近い点となり  $(g_0, g_j)$  又は  $(g_0, g_i, g_j)$  が出来る。3 番目に近い点は  $g_0, g_i$  又は  $g_j$  の何れかの neighbor のうちにある。……

ある step までに出来た tree node の集合を  $\{g_i | i \in I\}$ 、それらの neighbor 全体の集合を  $\{g_j | j \in J\}$  とする。いま  $g_i$  から  $g_j$  への辺の距離を  $c_{ij}$ 、tree に含まれる chain を通ってそれぞれの tree node  $g_i$  にいたる距離を  $C_i$  のようにあらわすとすると、各 step ごとに、

$$C = \min_i \min_j (C_i + c_{ij})$$

を計算することが必要である。 $C$  を最小にする  $i(j)$  を  $(i^*, j^*)$  とすると、点  $g_{j^*}$  と辺  $(g_{i^*}, g_{j^*})$  があらたに tree に加えられ、 $C$  が  $C_{j^*}$  となる。

こうして tree は step ごとに成長し、 $|G|$  ステップをへてネットワーク全体を span することになる。exampl 3 のネットワーク (第 5 図) における minimal tree の計算は第 3 表に示される。

第 3 表

step, tree nodes, neighbors: distance

0	$g(00)$	$g(01) : 5/6$ $g(05) : 7/6$ $g(14) : 2/6 *$
1	$g(00)$	$g(01) : 5/6$

	$g(14)$	$g(05) : 7/6$
		$g(02) : 2/6 + 2/6 = 4/6 \quad *$
		$g(13) : 2/6 + 7/6 = 9/6$
		$g(15) : 2/6 + 5/6 = 7/6$
2	$g(00)$	$g(01) : \min(5/6, 4/6 + 7/6) = 5/6 \quad *$
	$g(14)$	$g(05) : 7/6$
	$g(02)$	$g(13) : 9/6$
		$g(15) : 7/6$
		$g(03) : 4/6 + 5/6 = 9/6$
		$g(10) : 4/6 + 2/6 = 6/6$
3	$g(00)$	$g(05) : 7/6$
	$g(14)$	$g(13) : 9/6$
	$g(02)$	$g(15) : \min(7/6, 5/6 + 2/6) = 7/6$
	$g(01)$	$g(03) : 9/6$
		$g(10) : 6/6 \quad *$
4	$g(00)$	$g(05) : 7/6$
	$g(14)$	$g(13) : 9/6$
	$g(02)$	$g(15) : \min(7/6, 6/6 + 7/6) = 7/6 \quad *$
	$g(01)$	$g(03) : 9/6$
	$g(10)$	$g(04) : 6/6 + 2/6 = 8/6$
		$g(11) : 6/6 + 5/6 = 11/6$
5	$g(00)$	$g(05) : 7/6 \quad *$
	$g(14)$	$g(13) : 9/6$
	$g(02)$	$g(03) : \min(9/6, 7/6 + 2/6) = 9/6$
	$g(01)$	$g(04) : 8/6$

	$g(10)$	$g(11) : 11/6$
	$g(15)$	
6	$g(00)$	$g(13) : \min (9/6, 7/6 + 2/6) = 9/6$
	$g(14)$	$g(03) : 9/6$
	$g(02)$	$g(04) : \min (8/6, 7/6 + 7/6) = 8/6 \quad *$
	$g(01)$	$g(11) : 11/6$
	$g(10)$	
	$g(15)$	
	$g(05)$	
7 ~	省略	

---

tie-breaking rule : 先につくられた neighbor への route  
をより小さいとみなす。

## 文 献

1. Hu, T. C., Integer Programming and Network Flows, Addison-Wesley, 1969, 伊理正夫・今野浩・棚橋次彦訳, 整数計画法とネットワーク・フロー, 培風館, 1975. (第18, 19, 20章, 付録A)
2. Garfinkel, R. S., and G. L. Nemhauser, Integer Programming Wiley, 1972. (第6, 7章)
3. Salkin, H. M., Integer Programming, Addison-Wesley, 1975 (第9章)
4. 土倉幸子, 群の構造を用いるアルゴリズム, 長尾昭哉編, 行動選択のための推論, 恒星社厚生閣, 1973.
5. van der Waerden, B. L., Algebra II, Frederick Ungar, 1970, 第

5 版からの英訳 (第12章)。

6. ファン・デル・ヴェルデン, 現代代数学Ⅲ, 銀林浩訳, 東京図書, 1959,  
第2版からの訳, (第15章)。

(1979, 11, 26)