Ⅳ-91　最適スケジュールの決定法について

京都大学工学部　正員　吉川和広
京都大学大学院　学生員　○春名　攻

1. はじめに

近年、工事施工の分野においても計画・管理の科学化を目標としてかなりの努力が払われてきた。とくにPERT, CPM等のネットワーク・プランニング, スケジューリングの技法を中心として、数多くの実施計画が作成され、ある程度は実用に供するようになってきている。筆者等の研究グループも、PERT, CPMの土木工事への適用を行なうとともに、基礎データの収集や、計画作成のプロセス, システムの研究を行なってきたが、これら一連の事例研究をとおして、スケジュールにおける作業間の順序関係には①施工技術的な側面から先決的に定められる技術的な順序関係と②各種工事用資源の配分方法から定められる管理的順序関係の2通りのものが存在することを明らかにしてきた。従って、工事施工におけるスケジュールの問題は一般に、「①の技術的順序関係を与件とした場合、与えられた資源制約量の下で、プロジェクト完了時刻が最小になるような②の管理的順序関係すなわちスケジュールを求めよ。」のように表わされる。従来、この種の問題に対しては、主として実用的な側面からの要請に従って、比較的小さな完了時刻を与えると考えられる近似解をスケジュールとして求める手法としてのPERT/MANPOWERが開発されているにすぎない。しかし、列挙法によって最適解を求めた場合、これらの近似解と最適解の間に大きなへだたりが存在することが多い。従って、計算法が若干複雑になったとしても最適解法へのアプローチは是非とも行なっておく必要があると考える。本研究においてはこのような観点に立って、作業分割の可能な場合の最適スケジュール決定法の提案を行なうことにする。

2. 同時作業のパターンとスケジュール

上記の問題を数学モデルで表わすために、工事用資源が1種類の場合に限定してパターン, スケジュールの表現法についてまず述べることとする。いま、図-1に示したネットワークによって①の技術的な順序関係を表わすことにするが、これに対して、表-1のように、作業所要時間d_i, 所要資源量r_iおよび資源制約量Rを与える。このような条件の下で、1つの実行可能なスケジュールを求めたものを図-2のバーチャートに示した。このバーチャートにおいて作業の終了あるいは中断した時刻で時間軸を区分すると、時間区間$I_1, I_2, \cdots\cdots, I_5$が求められる。いま、作業$J_i$が区間$I_k$で実施されて

図-1

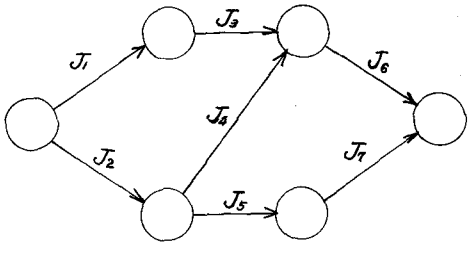


表-1　(制約量R=9)

J_i \ i	1	2	3	4	5	6	7
d_i	8	12	7	5	2	6	8
r_i	3	4	4	5	4	3	6

いるときに $a_{ik}=1$ とし、そうでない時には $a_{ik}=0$ とおくと、各区間において、同時刻に実施されている作業の組合せが求められる。このような作業の組合せをパターンと定義し、列ベクトル $\mathbb{P}_k$ で表わす。いま、図－2に示したスケジュールのパターンは表－2のようになるが、このようなパターンから構成されるマトリックスをスケジュールと定義し $\mathbb{B}$ で表わす。($\mathbb{B}=(\mathbb{P}_1\,\mathbb{P}_2\cdots\mathbb{P}_7)$)

3. パターンとスケジュールの実行可能性

以上のことから明らかなようにスケジュールは各区間に対するパターンの割当てによって規定される。もちろん、各パターンは制約条件

① $\quad \mathbb{r}\cdot\mathbb{P}_k \leq \mathbb{R}\ (k=1,2,\cdots,n)$ ここで、$\mathbb{r}=(r_1,r_2,\cdots,r_m)$, $\mathbb{R}=\begin{pmatrix}R\\ \vdots\\ R\end{pmatrix}\}m$ 個

を満していなければならない。また、1つのパターンに含まれる作業は同時作業が可能なものでなければならない。さらに、スケジュール $\mathbb{B}$ が実行可能であるためには与えられた技術的な順序関係に矛盾しないような作業順序を構成していなければならない。以下にパターンおよびスケジュールの実行可能性の判別の方法を示すこととする。

(1) パターンの実行可能性　パターンの実行可能性は上記①式の制約を満すことと同時作業が可能であることの2つの条件を満していなければならない。つぎに、同時作業の可能なパターンを判別するための条件について示す。いま、同時作業が可能な作業間の対応関係を表わすマトリックスを S_0 で表わす。S_0 は、

② $\quad S_0=(s^0_{ij})$, $s^0_{ij}=\begin{cases}1, & J_i \otimes J_j \ (\text{作業}J_i\text{と作業}J_j\text{の同時作業が可能})\\ 0, & \text{その他の場合}\end{cases}$

と表わされるが、s^0_{ij} は s^T_{ij} および $\bar{s}^T_{ij}$ を用いて、③ $s^0_{ij}=1-s^T_{ij}-\bar{s}^T_{ij}$ のように求められる。ここで s^T_{ij} および $\bar{s}^T_{ij}$ は技術的な順序関係およびそれに矛盾するような順序関係を表わしている。

④ $S_T=(s^T_{ij})$, $s^T_{ij}=\begin{cases}1, & J_i < J_j \left(\begin{array}{l}\text{作業}J_i\text{が作業}J_j\text{の先行作業}\\ \text{直接的な先行作業でなくてもよい}\end{array}\right)\text{の場合}\\ 0, & \text{その他の場合}\end{cases}$ ⑤ $\bar{S}_T=(\bar{s}^T_{ij})$, $\bar{s}^T_{ij}=\begin{cases}1, & s^T_{ji}=1\text{の場合}\\ 0, & \text{その他の場合}\end{cases}$

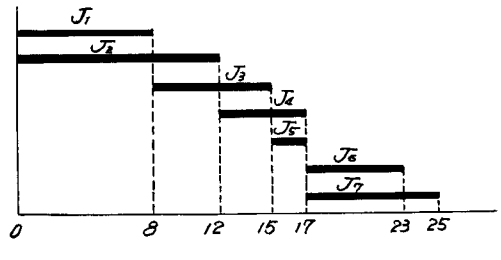
このように S_0 を定めると、実行可能なパターン $\mathbb{P}_k$ において、$a_{ik}=1$ であるような作業の集合を A_k とおき、つぎに、作業 J_i に対してマトリックス S_0 から $s^0_{ij}=1$ であるような作業 J_j の集合をとりだし、これを A_i とおくと、A_k と A_i の間には ⑥ $\bigcap_{J_i\in A_k} A_i \supseteq A_k$ が成立していなければならない。式⑥が実行可能なパターンの判別式である。(判別はカットセットによっても行なえるが、ここでは省略する。)

(2) スケジュールの実行可能性　パターン $\mathbb{P}_k$ を用いて表わされるスケジュール ⑦ $\mathbb{B}=(\mathbb{P}_1\mathbb{P}_2\cdots\mathbb{P}_m)$ から一意的に導びかれる順序関係および同時作業の対応関係を表わしたものを S とする。

表－2

区間	I_1	I_2	I_3	I_4	I_5	I_6	I_7
$\mathbb{P}_k$	1	0	0	0	0	0	0
	1	1	0	0	0	0	0
	0	1	1	0	0	0	0
	0	0	1	1	1	0	0
	0	0	0	0	1	0	0
	0	0	0	0	0	1	0
	0	0	0	0	0	1	1
x_k	8	4	3	0	2	6	2

図－2



⑧ $S=(s_{ij})$, $s_{ij}=\begin{cases}1, & J_i<J_j \text{ あるいは作業 } J_i \text{ と作業 } J_j \text{ が同時作業を行なっている場合.}\\ 0, & \text{その他の場合.}\end{cases}$

スケジュールBが実行可能であることは、S が S_T に矛盾していないことを示せばよい。これを調べるために、つぎのような作業間の関係を表わすマトリックス S_a を設定する。S_a は作業間の順序関係と同時作業関係のような許容的な関係をすべて表わしたもので、

⑨ $S_a=(s^a_{ij})$, $s^a_{ij}=\begin{cases}1, & J_i<J_j \text{ あるいは } J_i \otimes J_j \text{ の場合.}\\ 0, & \text{その他の場合.}\end{cases}$

のように示される。式②, ③から s^a_{ij} は次式によって求められる。⑩ $s^a_{ij}=1-\bar{s}^T_{ij}$ この S_a を用いるとスケジュールが実行可能であるための条件は、⑪ $\Delta S=S_a-S\geq 0$ (零行列) が成立することである。さて、ΔS が上式を満さないときには、現在のスケジュールは実行可能ではないが、スケジュールとパターンの関係からつぎの2つの場合が考えられる。ⓐ パターンの区間への割当てを適当に変更することによって $\Delta S=0$ とすることができる。ⓑ どのように割当てを変更しても $\Delta S\geq 0$ とすることはできない。ⓑの場合には、新しいパターンの組合せを考えなければ実行可能なスケジュールとはならないことを示している。

チ. 数学モデルと解法

(1) すべての実行可能なパターンが求められる場合. プロジェクトに含まれる作業の数が少ない場合には、あらかじめすべての実行可能なパターンを求めることは不可能ではない。いま、すべての実行可能パターンが求められているとき、モデルは、「制約条件 ⑫ $\sum_k P_k x_k=\mathbb{D}$, $x_k\geq 0$ の下で、目的関数 ⑬ $\lambda=\sum_k x_k$ を最小にするような解 $\{x^0_k\}$ を求めよ。ただし、ベイシスマトリックスBから求められるマトリックス S と S_a によって計算される ΔS は常に式⑪を満していなければならない。」のように表わされる。この定式化から明らかなように、後の条件がなければLPのモデルとなっている。従って、後の条件の存在しないときにはシンプレックス法によって解くことができるが、この条件に対してつぎのような方法で対処する。すなわち、新しい解 x_d の導入時に ΔS を調べ式⑪が成立しないとき、ⓐ, ⓑのいずれの場合かを判別し、ⓑの場合には $x_d=0$ とおいて、新しい解ベクトルを探せばよい。この場合の最適解の存在は容易に証明される。

(2) すべての実行可能なパターンが求められない場合 一般の場合のように、プロジェクトの作業の数が多くなると全体のパターンの数は幾何級数的に増大する。このため上記のような方法を採用することはできない。従って、ここでは必要最小限のパターン数を取扱うことによって最適解 $\{x^0_k\}$ を求める方法について述べることにする。まず、準備としてシンプレックス基準をつぎのように変更しておく。いま、C_k を目的関数の初期の係数、$\bar{C}_k$ を現在の係数、B をベイシスマトリックスとすると、$\bar{C}_k$ は、⑭ $\bar{C}_k=C_k-CB^{-1}P_k$, $C=(c_1, c_2, \cdots\cdots, c_n)$ として求められ、シンプレックス基準は、⑮ $\bar{C}_d=\min_k \bar{C}_k\ (<0)$ として求められる。ここで、式⑮に式⑭を代入すると、

⑯ $\bar{C}_d=\min_k\{C_k-CB^{-1}P_k<0\}=\max_k\{CB^{-1}P_k-C_k>0\}$ のように示されるから、シンプレックス基準をつぎのように変更することができる。すなわち、「まず、⑰ $U_d=\max_k\{CB^{-1}P_k\}$ によってインデックス d を求める。ここで、$U_d>C_d$ ならば解の改良が可能である。$U_d\geq C_d$ ならば現在の解が最適解を与える。」このように、U_d を用いてシンプレックス基準を表わすと、つぎのような

有利な点が考えられる。すなわち、1つの実行可能スケジュールが求められれば、BおよびB^{-1}が容易に計算され、新しく導入されるパターンR_kは式⑰によって求められるので、ピボット操作をすべてのパターンを対象にして行なう必要がなくなる。このことは、作業数が多くなればなるほど、全計算量を大幅に減少せしめることになる。さて、以上に計算に用いるパターン数の減少のための工夫について示したが、つぎに、式⑰を満すようなR_kの効率的な求め方について述べることにする。式⑰を書き改めると、「補助問題；制約条件 ⑱ $\sum_i R_{ki} y_i \leq R_k$, $y_i = 0$ or 1 の下で、目的関数 ⑲ $U = \sum_i \beta_i y_i$ を最大にするような $\{y_i^*\}$ を求めよ。ただし、解$\{y_i\}$は式⑥を満していることが必要である。」ここで、β_iは$\beta = CB^{-1} = \Pi \cdot B^{-1}$の要素であり、$R_k$は$R_k = y^0$, $y^0 = (y_1^0, \cdots, y_m^0)$として求められる。この補助問題は式⑤に関する条件さえなければ、Integer Programmingの問題であり、Gomory, Balas 等の解法で解くことができる。しかし、ここでは式⑥の条件があるためBalasの開発したAdditiveアルゴリズムに若干の変更を加えた解法を用いることにした。まず、以下で用いる記号を簡単に定義しておく。

A_λ；ノードλにおいて求められている解ベクトルy^λにおいて、$y_i^\lambda = 1$であるインデックスiの集合。 $A_\lambda = \{i ; y_i^\lambda = 1\}$

E_i；マトリックスS_0から求められるJ_iと同時作業可能な作業のインデックスの集合。 $E_i = \{k ; a_{ik} = 1\}$

G_λ；ノードλからのブランチによって求められるノードtにおいて新しく$y_i^t = 1$ ($y_i^t = 0$)とおく変数のインデックスiの集合 ($t = t_1, t_2, \cdots$)

M；$\beta_i > 0$であるインデックスiの集合。 $M = \{i ; \beta_i > 0\}$

R_k；資源kの制約量。

r_{ki}；資源kに対する作業J_iの所要量。

このように定義すると、目的関数の上界値（Upper Bound）は次式で表わされる。

⑳ $UB(A_t) = \sum_{i \in M \cap G_t \cup A_t} \beta_i$, $G_{At} = \bigcap_{i \in A_t} E_i - A_t$

$G_0 = \{i ; i = 1, 2, \cdots\cdots, n\}$

さらに、ブランチに際してのノードの評価には次式で与えるv_iを用いる。

㉑ $v_i = \begin{cases} \sum_{k} (R_k^{(\lambda)} - r_{ki}), & \text{if } R_k^{(\lambda)} - r_{ki} \geq 0, \text{ for all } k \\ -1, & \text{otherwise} \end{cases}$ $i \in G_\lambda$

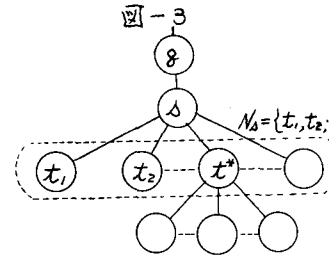
ここで、 $R_k^{(0)} = R_k$

$R_k^{(\lambda)} = R_k^{(\beta)} - r_{ki}$, $i \in G_\lambda$, $\lambda \in N_\beta$

このような上界値および評価値を用いた場合の補助問題の解法の手順は図－4のフローチャートのように示される。

主問題の計算法および補助問題の計算法の詳細は講演時にモデル計算例をもって、示すことにする。

図－3



図－4　補助問題 計算手順