経営科学 第11巻 第2号 (1968年1月)

プロジェクト・ネットワーク上での人員機械配分計画法 (第2部)†

山本正明*

§1. まえがき

第1部**においては，資源順序列を与えることによりプロジェクト・ネットワーク上に人員機械などの資源を配分する方法について述べた．第2部では，資源順序対 (resources order) を与える方法について検討することにする．

資源順序列は個々の資源について1本づつ与えられるので，1作業に必要な資源の数が増すと，プロジェクト全体で導入される順序関係の個数は非常に多くなり，ネットワークが複雑化する．そこで資源によって導入される順序関係を個々の資源ごとには考えず，総資源量の不足を解消するために必要となる順序関係だけをネットワーク {P} に加えていくことにする．これを資源順序対と呼ぶ．

総資源の過不足は時間とともに変動するから，日程上の時刻を追って共通の資源を競合する作業の集合を作り，その集合に属する作業の間に資源順序対を逐次的に与えていくことにより日程割付問題を解くことができる．

§2. コンフリクト作業集合による資源順序対の発生

次の条件を持つ問題について考える(1)．

1. プロジェクト・ネットワークは n 個の作業からなり，その先行作業行列{P}が与えられている．{P}はループを持たない．
2. 各作業の所要時間 d_j および作業に必要な資源量を与える資源行列{R}が与えられている．{R}の要素 r_{kj} は作業 j が資源 k を必要とする数量を示す．資源の種類は m 個あるから{R}は $m \times n$ 行列となる．
3. プロジェクト全体で利用可能な資源量を与える利用可能資源ベクトル{A}が与えられる．{A}の要素 a_k は資源 k の利用可能量を示し，全日程期間中一定のものとする．当然，$r_{kj} \leqq a_k$ である．
4. 上の条件の下でプロジェクト遂行期間 λ を最小化する．

---

† 1967年3月25日受理，1966年5月12日春季研究発表会に一部報告

* 法政大学工学部，

** 本誌第10巻3号 (1967年3月)

1) 記号は第1部と共通に用いてあり，その定義が第1部にあるものもある。

プロジェクトに属するすべての作業 j は最早開始時刻 ES_j に作業を開始するよう日程計画上に割付けられるものとする．したがって現在時刻 t が最早完了時刻 EF_j を過ぎるとこの作業の日程は確定したことを意味し，ネットワークから除外して考えてもよい．そこで時刻 t に対して未割付作業集合 $B(t)$ が定まる．

(2.1) $$B(t)=\{j|EF_j\geqq t\}$$

すべての先行作業を日程計画上に割付けられた作業は何時でも実行のできる作業である．そこで次のような実行待作業集合 $W(t)$ が定義できる．

(2.2) $$W(t)=\{j\in B(t)|{}^{\forall}i\in PJ_j, i\notin B(t)\}$$

$W(t)$ に属する作業に必要なすべての資源の合計が{A}の制限内ならば，全作業を日程計画上で実行できるが，もし

(2.3) $$\sum_{j\in W(t)} r_{kj} > a_k$$

となる資源 k があれば，すべての作業を実行するわけにはいかない．(2.3) 式を満たすような資源をコンフリクト資源と呼び，$\bar{k}$ で示す．$\bar{k}$ の制約で同時には実行できなくなった作業の集合をコンフリクト作業集合 $C_{\bar{k}}(t)$ と呼ぶ．

(2.4) $$C_{\bar{k}}(t)=\{j|r_{\bar{k}j}>0,\quad j\in w(t)\}$$

$C_{\bar{k}}(t)$ に属する作業は同時に実行できないから，そのうちの特定の作業 J を I の作業終了後に延期させることにする．すなわち

(2.5) $$I\ll J \qquad I,J\in C_{\bar{k}}(t)$$

なる順序関係を導入する．この順序関係は資源の制約から持込まれたものであるから資源順序対とよび，

(2.6) $$I\xrightarrow{\bar{k}}J$$

によって表示する[2]．

$C_k(t)$ に属する作業数と資源の制約との関連から，資源順序対を1個だけ導入したとしても，必ずしもコンフリクトは解消しないが，$C_k(t)$ から J を除いた新らしい $C_k'(t)$ についてさらに資源順序対 $I'\xrightarrow{\bar{k}}J'$ を指定し，順次 $J, J', J''\cdots\cdots$ を (Wt) から除外していくと，$C_k(t)$ は必ず解消する[3]．すべての資源について $C_k(t)$ が空集合となれば，その時 (Wt) に属するすべての作業は時刻 t における実行可能日程を作る．

時刻 t において資源 k に関するコンフリクト作業集合 $C_k(t)$ を解消するために導入された

---

2) 以後コンフリクト作業集合は，k の上の—をとり $C_k(t)$ であらわす．k がコンフリクト作業でなければ $C_k(t)$ は空集合と考える．

3) $C_k(t)$ 中の作業数が3個以上になると，2度目以降の $I'\longrightarrow J'$ によって，それ以前の $I\longrightarrow J$ が不要になる場合が起りうる．このため同時刻に発生した資源順序対の集合で不要になったものがないかを検討することが必要となる．この場合 k の検討の順序で結果が異る可能性も生ずる．§4の計算手順でも，この部分の補正は省略してある．

資源順序対は先行作業行列 $\{\hat{P}\}_{kt}$ で表わすことができる(4).

今 $t=0$ より始めて順次作業を割付けていき，全作業が割付けられたとすると，その過程で発生するすべての $C_k(t)$ から得られた $\{\hat{P}\}_{kt}$ の総和を考えると，これは資源の制約によってネットワークに持込まれる順序関係のすべてを与えている．したがって

$$\{P^*\}=\{P\}+\sum_t \sum_k \{\hat{P}\}_{kt} \tag{2.7}$$

は資源の制約を満足した実行可能日程である．$\bar{w}(t)$ を作る時の手順から $\{P^*\}$ にループの発生するおそれはない．

ネットワークに $I\longrightarrow J$ が加わると，作業 J の最早時刻は，

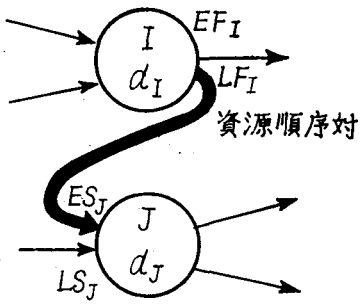
$$ES_J=EF_I, \qquad EF_J=EF_I+d_J \tag{2.8}$$

作業 I の最遅時刻は

$$LF_I=\min\ (LF_I^*,\ LS_J)$$
$$LS_I=\min\ (LS_I^*,\ LS_J-d_I) \quad \cdots(2.9)$$

に変更を受ける（第1図）．式中 * はこの資源順序対が加えられる前の日程時刻を示す．

(2.8)，(2.9) によって時刻を追いながら，順次各作業の日程時刻を修正していくことができる．



第1図　資源順序対の導入

§3. コンフリクト作業集合中での優先順位

$C_k(t)$ 中で $I\overset{k}{\longrightarrow}J$ を指定する方法は資源順序列を指定した時と同様に組合せ型の問題となる．したがってその指定法には種々の手段が考えられるが，まず優先順位法について検討してみる．$C_k(t)$ を作るために時間を追って各作業の ES_i を計算しているから，現在時刻 t までの日程計算の結果を，時刻 t の優先順位に反映させることができる．一般的にいえば，その方がより多くの情報が判断の材料として用いられているから，日程計算を始める前から確定している優先順位を用いるよりは，よい効果が期待できるはずである．

$I\overset{k}{\longrightarrow}J$ を指定する優先順位 PR は，作業対 $(I,\ J)$ を選択するものであるから，$C_k(t)$ に属するすべての作業対 $(i,\ j)$ について PR_{ij} が最も小さくなるような $(I,\ J)$ を採用することにする．第2，第3優先順位規則としては，必ずしも PR_{ij} 型のものでなくても，$(i,\ j)$ の一方だけを決める PR_i，PR_j 型の規則を用いてもよい．

第1部で述べたように，優先順位規則の有効性は実験的にしか判定できないものであり，目的

4)　$\{\hat{P}\}_{kt}$ の構造，加法については，第1部の $\{\bar{P}\}$ に準ずる．

に応じて自由に使って良いわけであるが，比較的有効と考えられるものを，以下に2～3列挙する．

〔A〕 プロジェクト遂行期間の増分 $\triangle\lambda$ を用いる方法

資源順序対 $I \xrightarrow{k} J$ をネットワークに付加することによるプロジェクト遂行期間の増分は次のように計算できる．

$I \longrightarrow J$ により，ここを通るパス $s \longrightarrow I$, $I \longrightarrow J$, $J \longrightarrow f$ が新らたに発生する．その長さ λ' は，

(3.1) $$\lambda' = EF_I + (\lambda - LS_J)$$

したがって増分 $\triangle\lambda$ は

(3.2) $$\triangle\lambda = \lambda' - \lambda = EF_I - LS_J$$

実際には λ の短縮はあり得ないから，

(3.3) $$\triangle\lambda = \max\ (0,\ EF_I - LS_J)$$

である．優先順位規則としては負の $\triangle\lambda$ も含めて，(3.2) 式を用いることにする．

(3.4) $$PR_{IJ} = EF_I - LS_J$$

PR_{IJ} が等しい場合の第2，第2優先順位としては

(3.5) $$PR_J^2 = 1/LS_J, \quad PR_J^3 = J, \quad PR_I^4 = I$$

を用いて (3.5) 式で定まる I, J を含む $I \longrightarrow J$ を採用する．

〔B〕 到着順優先サービス規則

〔A〕 では LS_J 時刻の遅いもの，すなわちプロジェクト完成までの残留作業時間の少ない作業を遅らせたが，資源への到着時刻すなわち ES_j の大きい作業を遅らせることによって，到着時刻の早い作業を優先的にサービスさせるようにすることができる．そこで，(3.4) 式の LS_J と ES_J とを入換えた．

(3.6) $$PR_{IJ} = EF_I - ES_J$$

を第1優先順位規則として用い，さらに

(3.7) $$PR_J^2 = d_J, \quad PR_J^3 = J, \quad PR_I^4 = I$$

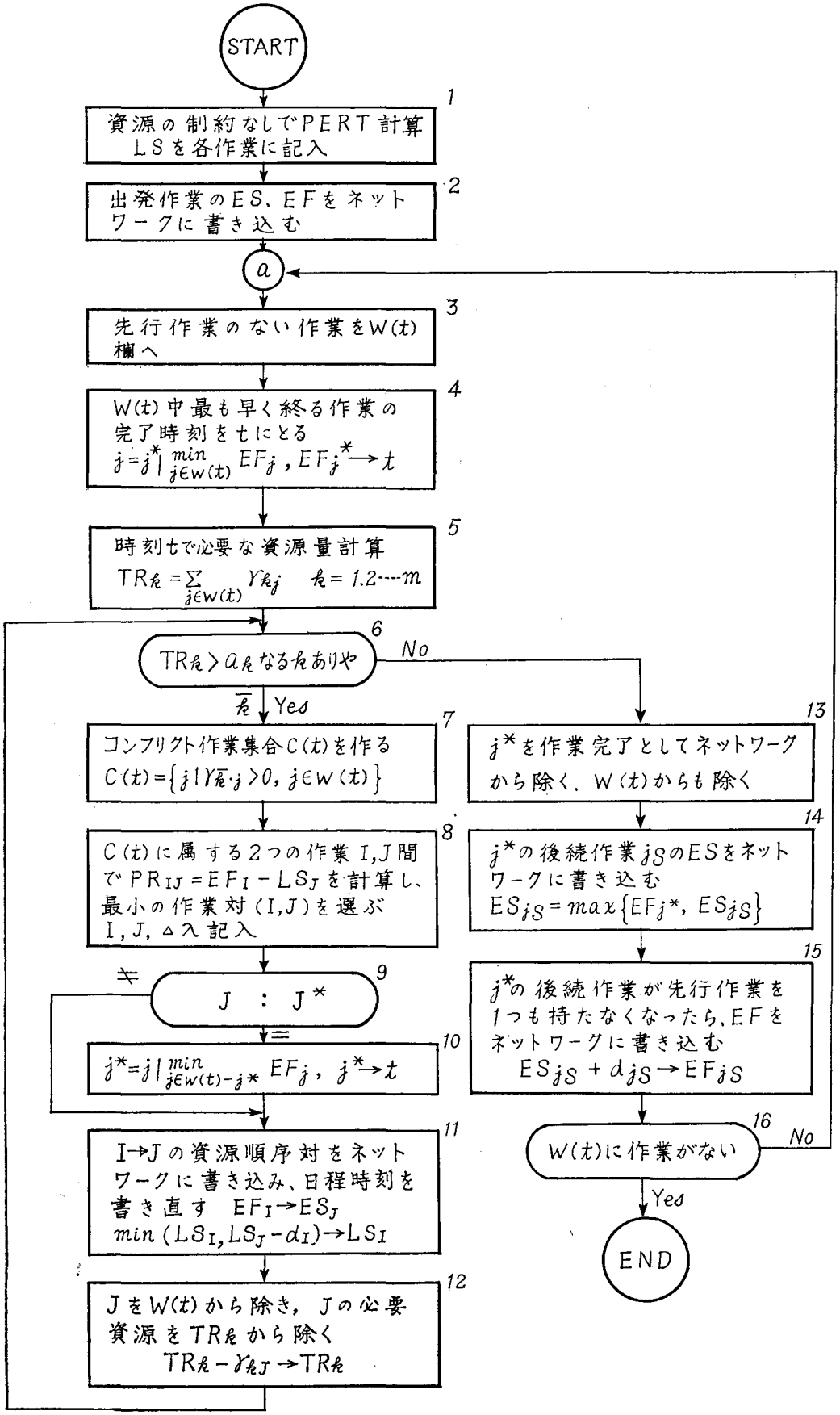
を付け加える．

〔C〕 重み修正 LS を用いる方法

〔A〕 の優先順位に入る LS_j は，作業 j 以後のプロジェクトの残留作業時間を，j の重要度を計る目安として用いたものである．しかもこれは日程計画の割付前から $\{P\}$ によって確定している値である[5]．そこで作業の重要度を与える他の目安があれば，これによって LS_j を修正して，これを優先順位に反映させることができる．

---

5) 最早日程は時刻 t とともに変化するが，最遅日程は $t=0$ で PERT 計算した値を最後まで使用する．たとえ再計算しても，全体に $\triangle\lambda$ が加算されるだけで，相対値は変らないからである．



第2図　資源順序対による日程計画計算手順

(3.8) $$\widetilde{LS}_j = LS_j\left(1+\frac{\bar{w}-w_j}{w_j}\times a\right)$$

w_j：作業 j の相対的重要度

$\bar{w}$：$C_k(t)$ に属する作業の w_j の平均値

a ：重み係数，$a \leqq 1$

w_j としてはたとえば，次のような諸量が利用できる．

1. 後続作業総数　　$j \prec h$ なる作業 h の総数
2. 後続作業総数＋(後続作業中の critical job の個数)$\times b$　　b：重み係数
3. 後続作業中の資源 k の総必要量

§4. 計 算 の 手 順

前節までの考え方にもとずき，時刻 t を 0 から次第に増しながら，順次必要な資源順序対を指定していくための手順を第 2 図に示す．実際の計算手順は FLOW 型 PERT 計算の基本アルゴリズムにそのまま乗るように組まれているが，第 2 図では直観的理解を助けるため手計算用に書きかえて，手順を一部変更してある．次に簡単な例題を示す．

〔例題 3〕

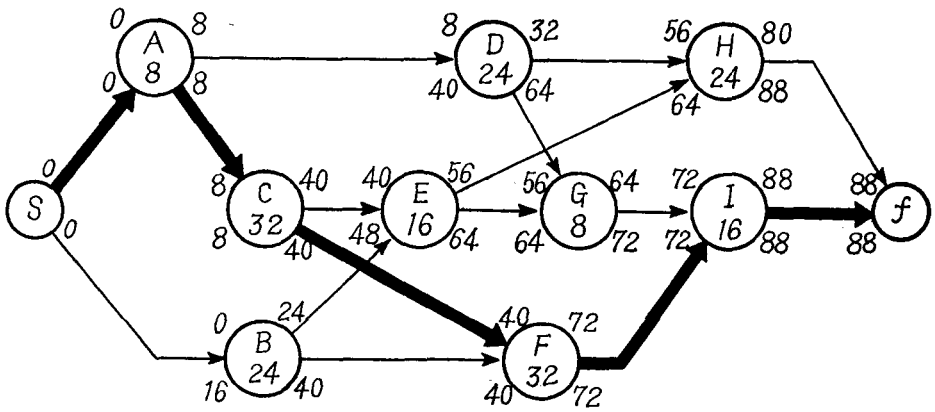
第 3 図に示すプロジェクト・ネットワークについて，日程計画を組め．所要時間リスト，資源行列，利用可能資源ベクトルは下に与えられている．

1) 所要時間リスト d_j

s	A	B	C	D	E	F	G	H	I	f
0	8	24	32	24	16	32	8	24	16	0

2) 資源行列 $\{R\}$ r_{kj}

k \ j											
I	0	5	3	4	5	7	2	5	4	4	0
II	0	0	6	0	1	6	5	5	3	0	0
III	0	1	3	3	1	5	5	0	1	2	0



第 3 図　資源の制約を入れない時の日程

3) 利用可能資源ベクトル$\{A\}$

a_k		
	I	8
	II	6
	III	6

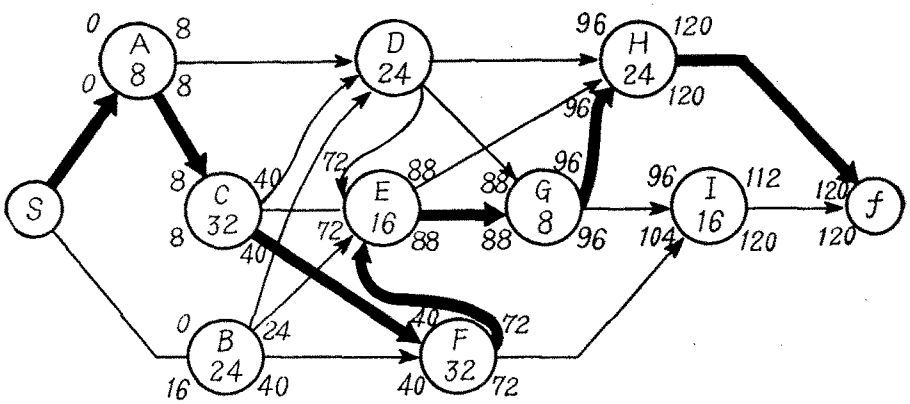
日程計算

優先順位規則としては〔A〕のプロジェクト遂行期間の増分 $\triangle\lambda$ を用いることにする．日程時刻はネットワーク上に資源順序対を書き込みながら計算し，第1表のようなコンフリクト表に結果を整理していくと，第2図の手順は簡単に実行できる．

第1表　コンフリクト表

レベル	t	j^*	$W(t)$	TR_k I	TR_k II	TR_k III	$C(t)$	I	J	$\triangle\lambda$	優先順位 No.	優先順位 EF	優先順位 LS
	0	s	$\overline{\text{s}}$	0	0	0							
	8	A	$\overline{\text{A}}$, B	8	6	4							
1	24	B	B, B, (D)	$\overline{12}$	7	7	B, C, D	B	D	0	B	24*	16
			$\overline{\text{B}}$, C	7	6	6					C	40	8
2	40	C	C, (D)	$\overline{9}$	1	4	C, D	C	D	0	D	32	40*
			$\overline{\text{C}}$	4	0	3					C	40*	8
3	56	E	D, (E), F	$\overline{14}$	12	11	D, E, F	D	E	16	D	48	40*
	64	D	$\overline{\text{D}}$, F	7	6	6					E	64*	40
4	72	F	(E), F	$\overline{9}$	11	10	E, F	F	E	24	F	72	40
			$\overline{\text{F}}$	2	5	5					E	80	48*
	88	E	$\overline{\text{E}}$	7	6	5					F	72*	40
5	96	G	G, (H)	$\overline{9}$	8	1	G, H	G	H	32	G	96*	64
			$\overline{\text{G}}$	5	5	0					H	112	64*
	112	I	H, $\overline{\text{I}}$	8	3	3							
	120	H	$\overline{\text{H}}$	4	3	1							
	120	f	$\overline{\text{f}}$	0	0	0							

註 1. レベルはコンフリクトの起った回数を示す．
2. $W(t)$ 欄中，—はその時刻に作業の完了したことを示す，また（ ）は資源順序対によって，その作業が $W(t)$ から外されたことを示す．
3. TR_k は資源別の総必要量を示し，—はコンフリクト資源となったことを示す．
4. 優先順位欄の＊印は $PR_{IJ}=EF_I-LS_J$ の最小となる (I, J) の組を示した．



第4図　資源順序対を加えた時の日程

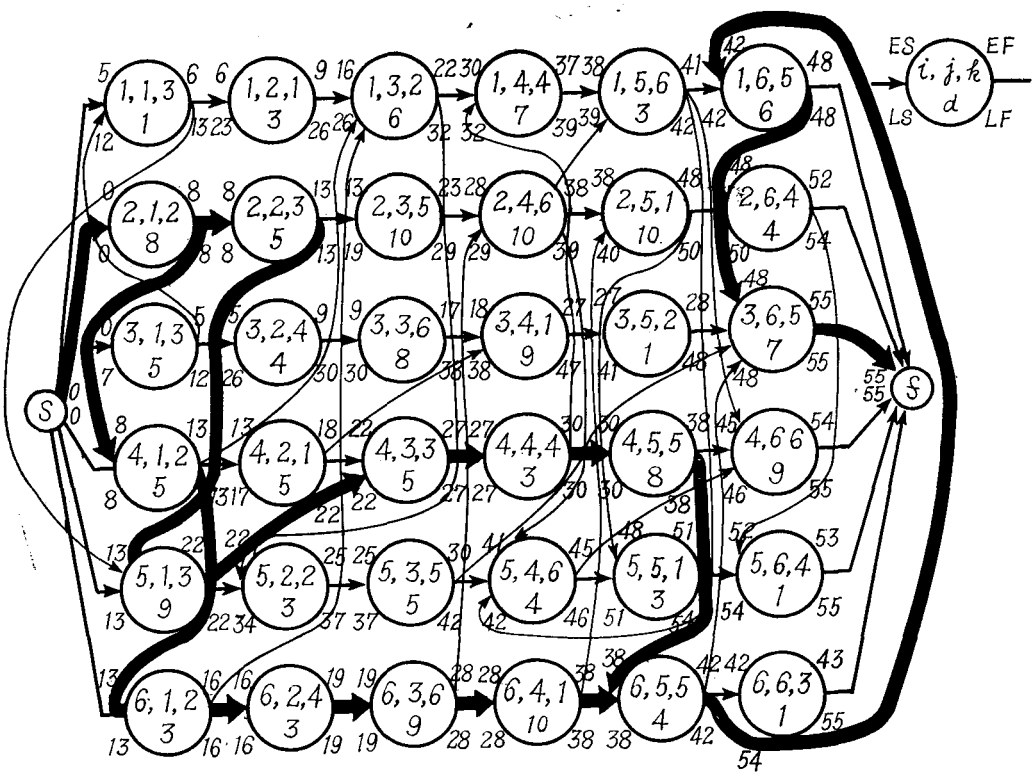
第2表　6×6×6 JOB SHOP 問題のコンフリクト表の一部

レベル	t	j^*	$W(t)$	TR_k						$C(t)$	I	J	$\triangle\lambda$	優先順位		
				I	II	III	IV	V	VI					No.	EF	LS
1	1	11	11, 21, 31, 41, 51, (61)		$\overline{3}$	3				21, 41, 61	41	61	0	21	8	0
														41	5*	12
2			11, 21, 31, (41) 51		$\overline{2}$	3				21, 41	21	41	0	61	3	17*
3			11, 21, 31, (51)		1	$\overline{3}$				11, 31, 51	11	51	0	11	1*	23
4			(11), 21, 31		1	$\overline{2}$				11, 31	31	11	0	31	5	3
														51	9	22*
	5	31	21, $\overline{31}$		1	1										
	6	11	21, $\overline{11}$, 32		1	1	1									
	8	21	$\overline{21}$, 12, 51, 32	1	1	1	1									
5	9	32	22, 41, 12, (51), 32	1	1	$\overline{2}$	1			21, 51	22	51	0	22	13*	8
														51	15	22*
			22, 41, 12, $\overline{32}$	1	1	1	1									
	9	12	22, 41, $\overline{12}$, 33	1	1	1			1							
6	13	22	22, 41, (13), 33		$\overline{2}$	1			1	41, 13	41	13	0	41	13*	12
														13	15	25*
			$\overline{22}$, 41, 33		1	1			1							
	13	41	23, 51, $\overline{41}$, 33		1	1		1	1							
7	16	61	23, 51, 42, 61, 33, (13)	1	$\overline{2}$	1		1	1	13, 61	61	13	0	13	19	25*
														61	16*	17
			23, 51, 42, $\overline{61}$, 33	1	1	1		1	1							
	17	33	23, 51, 42, 62, 13, $\overline{33}$	1	1	1	1	1	1							
8	18	42	23, 51, 42, 62, 13, (34)	$\overline{2}$	1	1	1	1		42, 34	42	34	0	42	18*	17
														34	26	30*
			23, 51, $\overline{42}$, 62, 13	1	1	1	1	1								
9	19	62	23, 51, 34, (43), 62, 13	1	1	$\overline{2}$	1	1		51, 43	51	43	0	51	22*	22
														43	23	22*
			23, 51, 34, $\overline{62}$, 13	1	1	1	1	1								
	22	51	23, $\overline{51}$, 34, 63, 13	1	1	1		1	1							
10	22	13	23, (52), 43, 34, 63, 13	1	$\overline{2}$	1		1	1	13, 52	13	52	0	13	22*	25
														52	25	31*
			23, 43, 34, 63, $\overline{13}$	1	1	1		1	1							
	23	23	$\overline{23}$, 43, 34, 63, 52, 14	1	1	1	1	1	1							
11	25	52	(24), 52, 43, 34, 14, 63	1	1	1	1		$\overline{2}$	24, 63	63	24	5	24	33	23*
														63	28*	23
			$\overline{52}$, 43, 34, 14, 63	1	1	1	1		1							
	27	34	53, 43, $\overline{34}$, 14, 63	1		1	1	1	1							
	27	43	53, 43, 35, 14, 63		1	1	1	1	1							
12	28	63	53, 44, 35, (14), 63		1		$\overline{2}$	1	1	44, 14	44	14	0	44	30*	27
														14	29	31*
			53, 44, 35, $\overline{63}$		1		1	1	1							
	28	35	53, 44, $\overline{35}$, 24, 64	1	1		1	1	1							
13	30	53	53, 44, (36), 24, 64	1			1	$\overline{2}$	1	53, 36	53	36	0	53	36*	34
														36	35	40*
			$\overline{53}$, 44, 24, 64	1			1	1	1							
14	30	44	(54), 36, 44, 24, 64	1			1	1	$\overline{2}$	24, 54	24	54	0	24	38*	23
														54	34	29*
			$\overline{44}$, 36, 24, 64	1			1	1	1							
15	37	14	14, 45, 36, 24, 64	1			1	$\overline{2}$	1	45, 36	45	36	0	45	38*	30
														36	37	40*

最終結果のネットワークは第4図のようになり，5本の資源順序対が付け加わったことになる．

1作業1機械，利用可能機械1台の型の JOP SHOP 問題にも，この手順はそのまま適用できる．第1部で用いた 6×6×6 の〔例題2〕について，資源順序対を入れた結果は第5図のようになる．第2表はそのコンフリクト表の1部である．

第5図から解るように $\lambda=55$ で全日程が割付けられた．(この値はこの問題では最適解である)．



第5図　6×6×6 JOP SHOP 問題の資源順序対

§5. 問題の拡張

1. 利用可能資源量{A}が時間とともに変動する場合

{A}が時間の関数として与えられる場合にも前節の計算手順を多少変更すれば，そのまま使用できる．たとえば，1日3交代制で，交番ごとに利用できる資源量が変動する場合や，あらかじめ増員計画が決められていて，それに合わせて，日程計画を組む場合などである．

利用可能資源ベクトル{A}は時間とともに変化するので行列$\{A\}_{kT}$と表わすことができる．ここで T は利用可能資源量の変化する期間につけられた期間番号で，必ずしも等間隔にとられている必要はない．

各時刻での利用可能資源量の変化を示すため，次のようなダミー作業を考える．

1. $\{A\}_{kT}$ の各列に対して，ダミー作業 $A(T)$ を考える．$A(T)$ の所要時間は第 T 期目の長さとする．
2. $A(T)$ はその実行にすべての資源を必要とするが，実際の必要量は0である．

$$r_{k\cdot A(T)}=0$$

3. 日程計算では最早時刻は通常の作業と同じ規則に従うが，最遅開始時刻は常に0とする．
4. $W(t)$ の中には必ず1個 $A(T)$ が存在し，これに対応する $\{A\}_{kT}$ の列がその時刻での利

用可能資源ベクトルを示す．

このダミー作業を期間順に直列に並べて，開始作業と終了作業の間を結ぶ．ただしこのダミー作業のパスを付加する時点ではプロジェクトの長さは未定であるから，ダミー作業パスの長さは，プロジェクトの長さだけあることに仮定しておく．このネットワークについて，前節の手順を適用して日程計算を実行する．利用可能資源量は，$W(t)_1^{\prime}$ に存在する $A(T)$ によって定まり，$A(T)$ が j^* に入るごとに$\{A\}$が変ることになる．また $A(T)$ はすべてのコンフリクト作業集合にも必ず1個入り，$A(T)\xrightarrow{k}J$ の形の資源順序対を作ることも起こる（$I\xrightarrow{k}A(T)$は起こりえない）．これは現在時刻に最も近い利用可能資源量が変化する時刻まで J の実行を延期することを意味する．

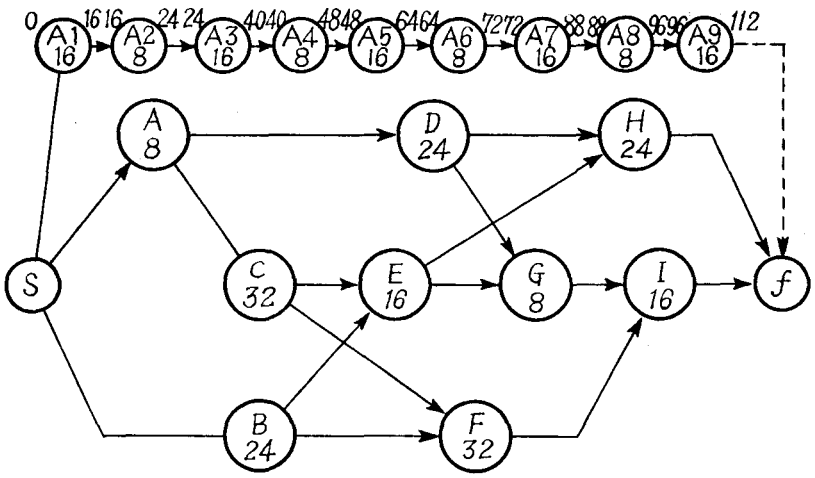
前節の例題を1日3交代で$\{A\}$が変わるものとして日程計画を作ってみよう．

	時間 → 0～8	8～16	16～24	⟶以下24時間周期の繰返し
I	10	10	8	
II	8	8	6	
III	8	8	6	

$\{a\}_{kT}$ で表わすと

$$\begin{Bmatrix} 10 & 8 & 10 & 8 & \cdots\cdots \\ 8 & 6 & 8 & 6 & \cdots\cdots \\ 8 & 6 & 8 & 6 & \cdots\cdots \end{Bmatrix}$$

となる．列の数はプロジェクトの長さだけ続くものとする．ダミー作業のパス $A(1)$, $A(2)$,… を第3図のネットワークに書き入れると第6図が得られる．



第6図　ダミー作業パスを付加したネットワーク

このネットワークについて日程計算を行なうと第3表のようなコンフリクト表が得られる．レベル1，4のコンフリクトでは$A(T)\longrightarrow J$ 型の資源順序対が発生し，これが資源量の変動によ

第3表 {A}が変化する場合のコンフリクト表

レベル	t	j^*	a_k I	II	III	$W(t)$	TR_k I	II	III	$C(t)$	I	J	$\triangle\lambda$	優先順位 No.	EF	LS
	0	s	0	0	0	$\overline{s}$	0	0	0							
	8	A	10	8	8	$\overline{A}$, B, A 1	8	6	4					B	24	16
														C	40	8
1	16	A 1				B, C, (D), A 1	$\overline{12}$	7	7	B, C, D, A 1	A 1	D	0	D	32	40*
						B, C, $\overline{A\,1}$	7	6	6					A1	16*	0
														B	24*	16
2	24	B, A 2	8	6	6	B, C, (D), A 2	$\overline{12}$	7	7	B, C, D, A 2	B	D	0	C	40	8
						B, C, $\overline{A\,2}$	7	6	6					D	40	40*
														A2	24	0
	40	C, A 3	10	8	8	D, $\overline{C}$, $\overline{A\,3}$	9	1	4					D	48*	40
														E	57	48*
3	48	D, A 4	8	6	6	D, (E), F, A 4	$\overline{14}$	12	11	D, E, F, A 3	D	E	0	F	72	40
						$\overline{D}$, F, $\overline{A\,4}$	7	6	6					A4	48	0
														E	64*	48*
4	64	E, A 5	10	8	8	(E), F, A 5	9	$\overline{11}$	10	E, F, A 5	A 5	E	16	F	72	40
		A 5				F, $\overline{A\,5}$	2	5	5					A5	64	0
														E	80	48*
5	72	F, A 6	8	6	6	F, (E), A 6	9	$\overline{11}$	10	E, F, A 6	F	E	24	F	72*	40
						$\overline{F}$, $\overline{A\,6}$	2	5	5					A6	72	0
	88	E, A 7	10	8	8	$\overline{E}$, $\overline{A\,7}$	7	6	5							
														G	96*	64
6	96	G, A 8	8	6	6	G, (H), A 8	$\overline{9}$	8	1	G, H, A 8	G	H	32	H	112	64*
						$\overline{G}$, $\overline{A\,8}$	5	5	0					A8	96	0
	112	I, A 9	10	8	8	$\overline{I}$, H, $\overline{A\,9}$	8	3	3							
	120	H, A10	8	6	6	$\overline{H}$, $\overline{A10}$	4	3	1							
	120	S				$\overline{f}$	0	0	0							

る制約を日程に与えてている（実際にはその後に発生した $I \longrightarrow J$ によって制約としてきいてこない）.

2) 作業の分割が許される場合

前節の計算手順では，一たん作業が開始されれば，途中での中断は許されないことを前提にしている．このため資源に遊びができてその利用率が下る場合も起こってくる．もちろん作業によってはコストの面から，あるいは技術的理由から中断の許されない作業も多いが，分割可能な作業があれば必要に応じて自由に作業を分割して日程計画を組んだ方がプロジェクト遂行時間の短縮の可能性が生れる．

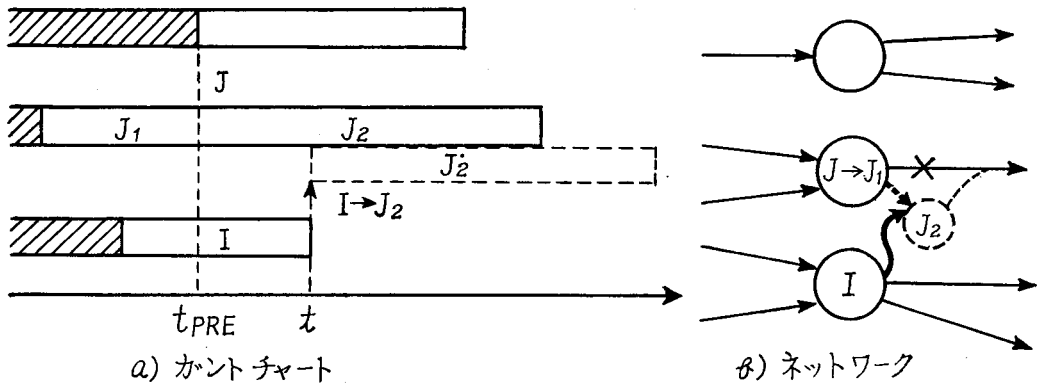
第7図の(a) ガント・チャートにおいて，$I \longrightarrow J$ なる資源順序対によって J を I の後に移動させる場合，すでに{A}との検討が完了している t_{PRE} より前に作業すべき J_1 の部分は，そのまま残しておいて，残りの部分 J_2 だけを $I \longrightarrow J_2$ として移動させることにする．ここで t_{PRE} とは現在時刻 t の1回前に $W(t)$ を作って検討した時刻である．

同様な操作をネットワーク上で考えると，第7図 (b) となる．この場合は新しい作業 J_2 をネットワークに書き足して，それとの間に $I \longrightarrow J_2$ を書けばよい．

この計算のために必要な第2図の手順の細かな変更は省略するが，

1) 作業 J が作業分割を許される．

2) $t_{PRE} > EF_J - d_J$



第7図　作業の分割が許される場合の日程

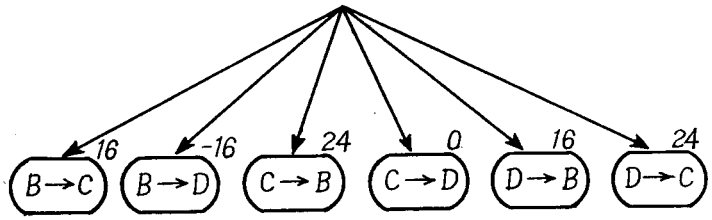
の条件が満たされる作業 J については，優先順位の計算に最遅時刻を用いる場合に

$$LS_J = LS_J^* - (t_{PRE} - ES_J)$$

を用いる．* は作業分割を考えない時の日程時刻を示す．もしこの作業が $I \longrightarrow J$ の中に入った場合は，第7図のような作業の分割をネットワークに考えて，$I \longrightarrow J_2$ の資源順序対を導入すればよい．

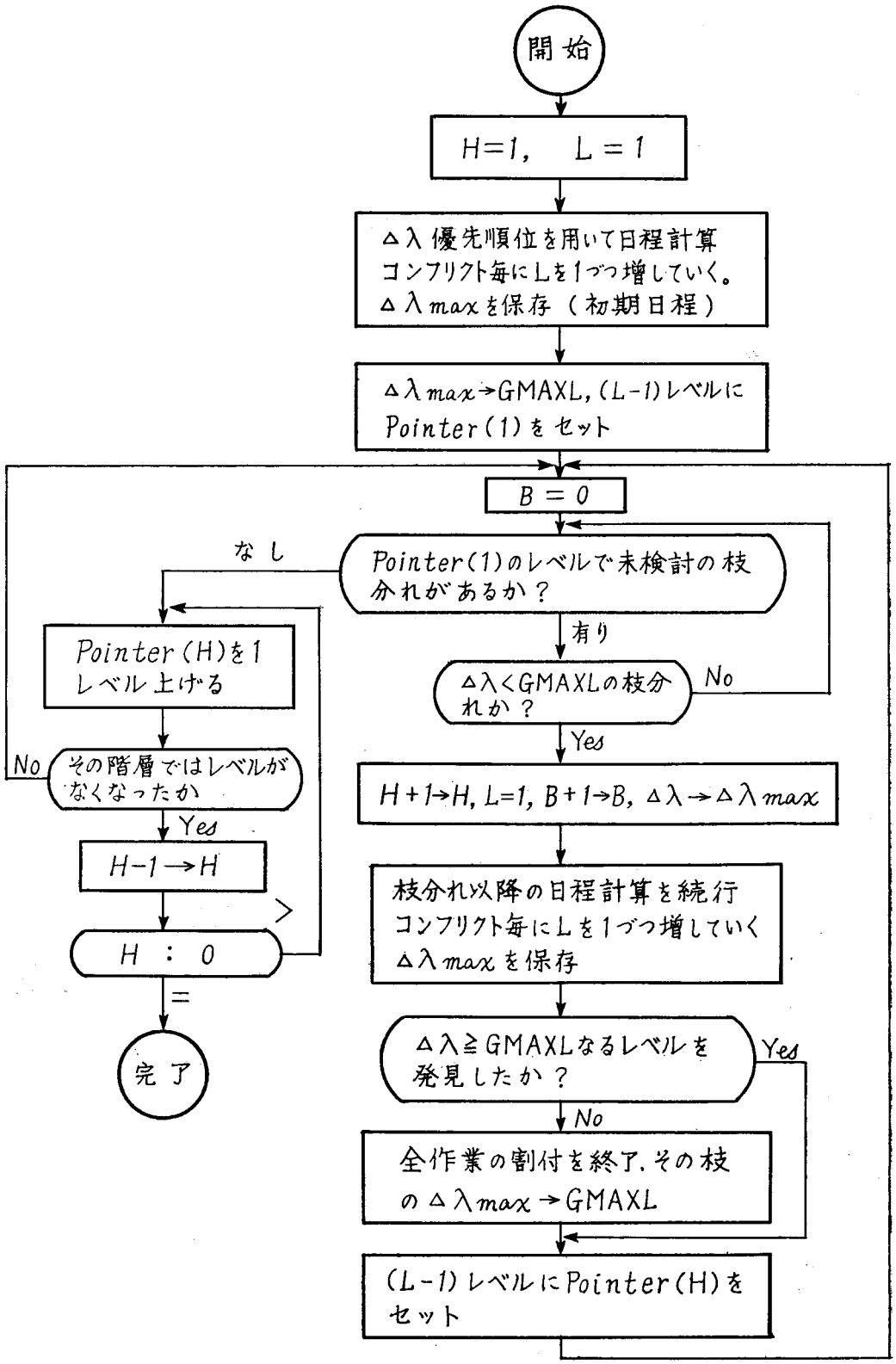
§6. Branch-and-Bound 法による最適日程の探索

コンフリクト作業集合に属する n ケの作業の間にとりうる資源順序対の数は $n(n-1)$ 組である．前節までは便宜的に PR_{IJ} を最小にするものを選んだが，これが最適日程を得るための最良の選択であるか否かその時点で決める手段はない．〔例題3〕のコンフリクト表でレベル1では第8図に示すような6本のとり得る枝がある．右肩に書いた数値は $\triangle\lambda$ を示すが，§4では $\triangle\lambda$ の最も小さい $B \longrightarrow D$ を選んだのである．同様な枝分れはレベル2でも生ずる．このような



第8図　レベル1での資源順序対選択

枝分れによって，レベル1を頂点とした．資源順序対の選択に関する tree が形成される．これを scheduling tree と呼ぶことにする．scheduling tree を頂点から末端までたどる1本の線は，その線上にある資源順序対をもつ1つの実行可能日程 $\{P^*\}$ に対応している．したがって scheduling tree をたどってすべての日程を計算すれば，その中から最適日程を決めることができる．この scheduling tree をたどる操作をできるだけ能率よく実行するため，branch-and-



第9図　branch and bound 法のフロー・チャート

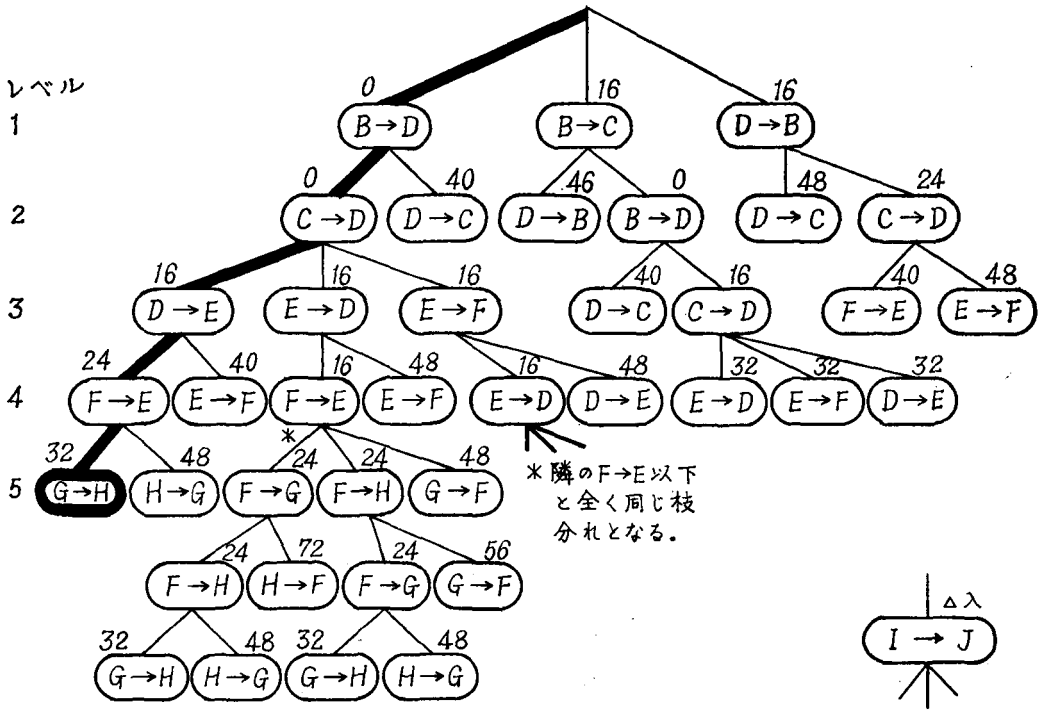
bound 法を利用する．

branch-and-bound 法の lower bound としては $\triangle\lambda$ 優先順位そのものを用いると便利である．§3 で説明したように，$\triangle\lambda$ 優先順位

$$PR_{IJ}=EF_I-LS_J$$

は新らたに導入された $I\longrightarrow J$ によるプロジェクト遂行時間 λ の増分である．資源の制約なしでのプロジェクト遂行時間を λ_T とすると，新らたに $I\longrightarrow J$ を加えられたネットワークのプロジェクト遂行期間は $(\lambda_T+\triangle\lambda)$ となる．scheduling tree を頂点から末端までたどる1本の線の上で，この線の表わす日程計画のプロジェクト遂行時間 λ は，各 $I\longrightarrow J$ での $\triangle\lambda$ のうち，最も大きい値 $\triangle\lambda_{max}$ によって決められる．最適日程として λ 最小のものを選ぶのであるから，ある日程計画のもつ $\triangle\lambda_{max}$ より大きい値の $\triangle\lambda$ が scheduling tree 上に現われたら，その点を通る枝で与えられる日程はすべてはじめのものより λ が大きくなり，それ以降の検討は無意味となる．§4 の手順では各レベルで $\triangle\lambda$ 最小の枝をたどって日程を求めているから，この道の持つ $\triangle\lambda_{max}$ は比較的他の枝の $\triangle\lambda_{max}$ よりは小さくなっているはずである。したがって他の枝の大部分は途中でこれより大きい $\triangle\lambda$ が現われ，それ以降の計算が省略できる可能性が強い．scheduling tree を能率的に探索する手順は第9図のフロー・チャートに示す．

枝分れの数を減らすために，$I\longrightarrow J$ のうち J の共通なものは，最も $\triangle\lambda$ の小さいものだけを用いることにする。資源順序対では，どの作業を遅らせるかという点に本質的意味があるので



第10図 〔例題3〕の scheduling tree

第4表 〔例題3〕の板分れ表

H＝1			H＝2			H＝3				I	J	$\triangle\lambda$	$\triangle\lambda_{max}$	GMALX	
P(1)	L	L	P(2)	L	B	P(3)	L	B							
12→	1	0								B	D	0	0		初期日程
11→	2									C	D	0	0		
2→	3									D	E	16	16		
1→	4									F	E	24	24		
	5									G	H	32	32	32	
	3	1								E	D	16	16		
			6→	1	0					F	E	16	16		
			4→	2						F	G	24	24		
			3→	3						F	H	24	24		
				4						G	H	32	32		
				2	1					F	H	24	24		
						5→	1	0		F	G	24	24		
							2			G	H	32	32		
	3	2								E	F	16	16		
			10→	1	0					E	D	16	16		
			8→	2						F	G	24	24		
			7→	3						F	H	24	24		
				4						G	H	32	32		
				2	1					F	H	24	24		
						9→	1	0		F	G	24	24		
							2			G	H	32	24		
	1	1								B	C	16	16		
			14→	1	0					B	D	0	16		
			13→	2						C	D	16	16		
				3						E	D	32	32		
	1	2								D	B	16	16		
			15→	1						C	D	24	24		
										F	E	40	40		

註1） H：階層番号，P(1)：pointer，L：レベル番号，B：枝分れ番号，$\triangle\lambda_{max}$：H一定の枝の中での $\triangle\lambda$ の最大値，GMAXL：全日程中での $\triangle\lambda$ の最大値．

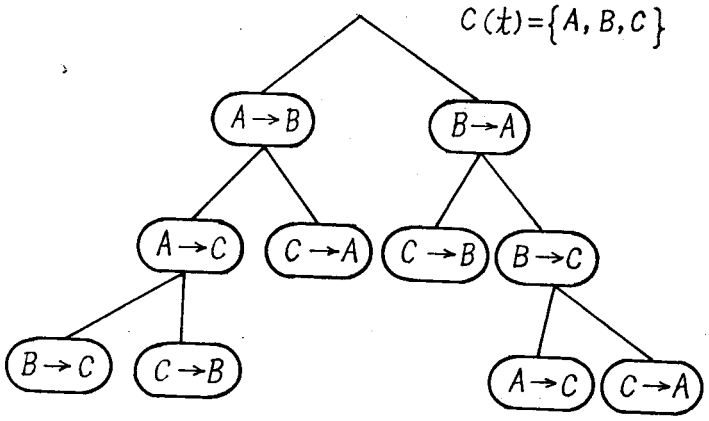
註2） P(H) 欄中の数字は pointer が移動する状態を示すために入れた一連番号．

註3） 実際の計算ではコンフリクト表と板分れ表とを同時に記入できる表を使うと便利である．

あって，どこまで遅らせるかは一時的に EE_I の最も小さい作業の後に続けておいてもよい．もしさらに遅らせる必要があるのなら，次のコンフリクト作業集合の中で処理できるからである．こうすれば枝分れの数は $n(n-1)$ 本から n 本に減少する．〔例題3〕に第9図の手順を適用した結果を枝分れ表としてまとめておく（第4表）．これで scheduling tree を書くと第10図のようになる．図中太線の日程が $H=1$ の初期日程であり，これが結局最適日程として最後まで生き残っている．

§7. JOB SHOP 型問題の最適日程

scheduling tree の枝分れの数は，$C(t)$ に属する作業数となるから一定とならない．しかし利用可能機械1台，1作業1機械の JOB SHOP 型の問題では，たとえ $C(t)$ に何個作業が来ても，2作業間だけでの順序関係 $I \longrightarrow J$, $J \longrightarrow I$ についての選択によって scheduling tree を作っても差支えない．$C(t)$ は最終的には1作業だけの集合にしなければならないのだから，2本だけの枝分れをつなげることによりすべての場合を尽せるからである（第11図）．



第11図　JOB SHOP 型問題の枝分れ

この場合は scheduling tree の枝分れ数は2で固定となり，第9図の branch-and-bound の手順は枝分れ番号 B が1までしかとり得ないことから，かなり簡素化できる．

さらに利用可能機械1台という制約から，branch-and-bound に用いる lower bound を改善することができる．これまでの $\triangle\lambda$ はネットワークの技術的順序関係による最遅時刻から計算したが，特定の機械に対する残留機械時間がその機械を必要とする作業の最遅時刻に対する制約としてきいてくることを考慮する．いま機械 k を必要とする作業の集合を JM_k とすると，時刻 t における機械 k に対する残留作業集合 $RM_k(t)$ は

$$RM_k(t)=\{j \mid j\in B(t),\ j\in JM_k\} \tag{7.1}$$

となる．$RM_k(t)$ を用いて，残留機械時間

$$\widehat{D}_k(t)=\sum_{j\in RM_k(t)} d_j \tag{7.2}$$

が各時刻ごとにすべての機械について計算できる．そこで機械 k の残留機械時間から，作業 i に与えられる最遅日程は，

$$\widehat{LS}_{ik}=\lambda-\widehat{D}_k(ES_i) \tag{7.3}$$

となる．

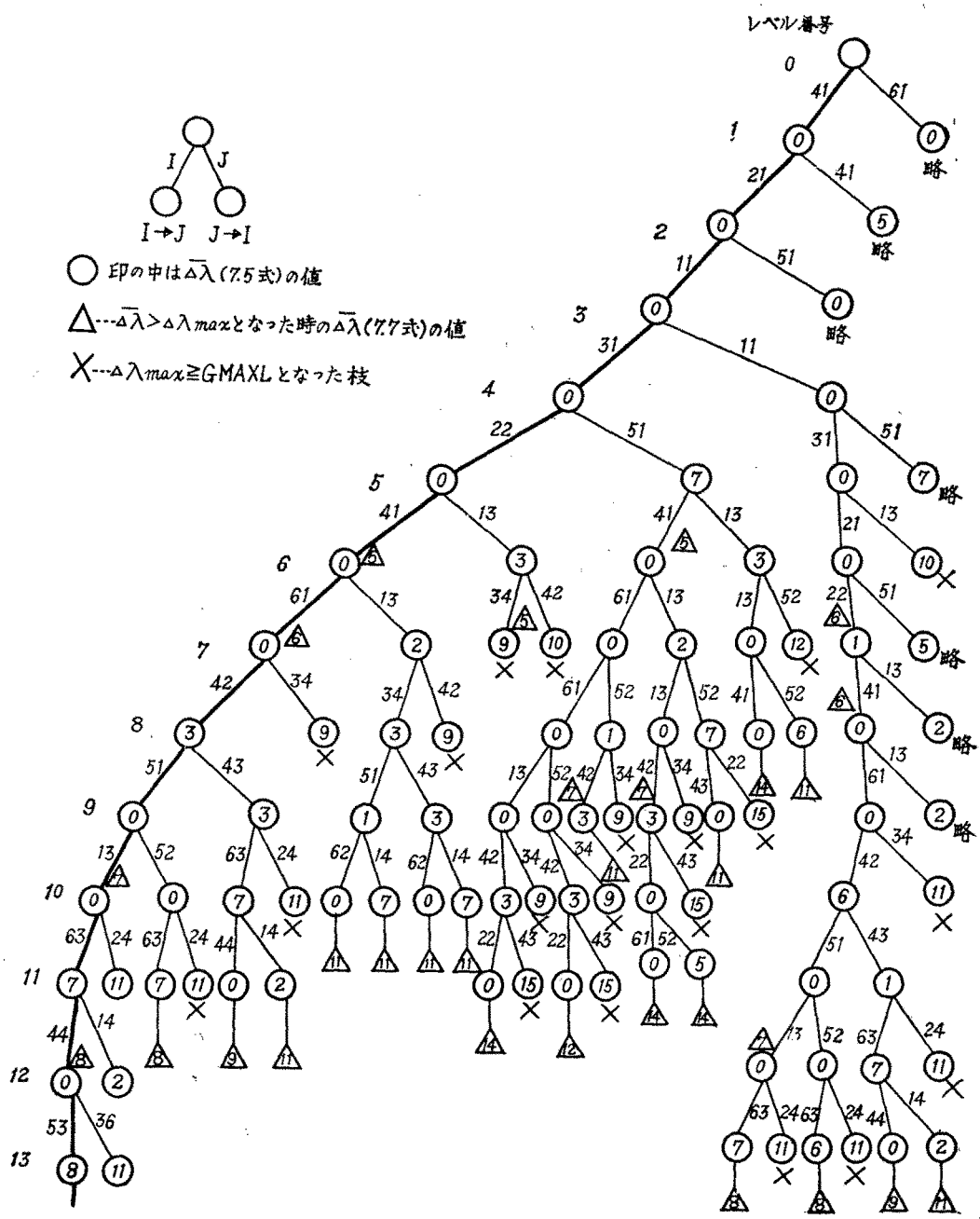
$I \longrightarrow J$ が導入された時，従来の $\Delta\lambda$ の外に (7.3) 式の最遅時刻も考慮して，

(7.4) $$\overline{LS}_J=\min\ (LS_J,\ \widehat{LS}_{Ik}+d_I)$$

を用いて $\triangle\lambda$ を計算する．すなわち

(7.5) $$\triangle\overline{\lambda}=EF_I-\overline{LS}_J=\max\ \{\triangle\lambda,\ EF_I-(\widehat{LS}_{Ik}+d_I)\}$$

$\triangle\overline{\lambda}$ を新しい lower bound として用いることができる．



第12図　6×6×6 JOB SHOP 型問題の scheduling tree の一部

つぎに，技術的順序関係 $i \ll j$ がある時，作業が完了して時刻 EF_i に作業 j が $W(t)$ に入ったとすると，$\widehat{D}_k(t)$ の制約から，

(7.6) $\quad \widehat{LS}_{jk}=\lambda-\widehat{D}_k(ES_j)$

が導入される．そこで

(7.7) $\quad \triangle\widehat{\lambda}=ES_j-\widehat{LS}_{jk}$

によって作業 j が ES_j に割付けられたことによる λ の増分が与えられる．この $\widehat{\triangle\lambda}$ は資源順序対の選択時の枝分れではなく，枝の途中で日程に持込まれるものであるが，やはりこの日程の lower bound となる．

ある特定の機械が日程上の隘路となって，プロジェクト遂行時間が延びるような場合には，残留機械時間による $\triangle\lambda$ の修正は非常に有効にきいてくる．

これらの lower bound を用いて〔例題2〕の 6×6×6 JOB SHOP 型問題を解いた scheduling tree の一部を第12図に示す．左側の太線が $\triangle\lambda$ 優先順位規則で求めた初期日程で，レベル25で全作業の割付を完了している．この線上の $\triangle\lambda_{max}$ は8である．図で解るようにレベル12ですでに $\widehat{\triangle\lambda}=8$ が現われ，計算量はかなり大巾に節約されている（$\triangle\lambda=8$ が現われるのはレベル23である．）

§8. 手法の特徴と実用上の問題点

第1部・第2部を通して順序指定型の人員機械配分計画法について述べてきた．これを要約すれば，プロジェクト・ネットワーク中の作業間の順序関係を，プロジェクトに固有の技術的順序関係と，日程計画者の意志により選択できる資源による順序関係とに区別し，後者に対する適切な選択方法を与えることによって資源配分問題を解決しようとするものであった．

1. 順序指定型の中での比較

第1部で述べた資源順序列を与える方法では，個々の資源についての作業順序を指定していくため同一種類の資源の個数は1，多くても数個までが限度である．したがって JOB SHOP 型の問題や，日程計画にクリティカルにきく特定の機械設備だけの配分を検討する場合には有効であるが，プロジェクトの総必要人員の配分計画というような形の問題に対しては，第2部の資源順序対を指定する方法を用いねばならぬ．この方法が JOB SHOP 型の問題にも有効であることは§7に述べた．

次に計算量の問題であるが，当然のことながらより正確な解を求めようとすれば計算量は増加する．第1部4.1の優先順位法は最も計算が簡単である．すでにプロジェクト・ネットワークが図面で与えられているような場合，特定の機械設備についての配分計画をネットワーク上に資源順序列を書き込みながら手計算で検討できる．それ以外の方法は計算機を利用しなければ，実用的には問題にならない．資源順序列の選択では，許される計算量にみあった選択方法を採用でき

る点が有利である．資源順序対の選択は前者に比して計算量は多くなる．特に branch-and-bouhd 法では，全 scheduling tree の探索に要する計算量が問題の形に大きく影響され，かつ膨大となるから，十分注意する必要がある．

2. 従来の資源配分法との比較

JOB SHOP 型の問題に対する日程計画法には従来非常に多くの研究があるが，これらの諸手法に対する順序指定型の配分法の最も大きい特徴は，日程計画がネットワーク上の順序関係で得られる点にある．PERT 型の日程計算を行なう操作はガントチャート上の時点に個々の作業を割付けることと本質的には同じことをやっているのであるが，マスタープランがネットワーク上に作られている点は実用上の利点が大きい．PERT/CPM の手法はプロジェクト管理の手法としてその優位性が明らかとなっているが，その中で開発された種々の日程管理上の手段を JOB SHOP 型の日程計画に応用する可能性が生れる．

〔例題2〕の6×6×6 の問題は H. Fisher, G. L. Thompson 等が用いた例題であるが[1]，5000のアクティブ・スケジュールをモンテカルロ法によって作った場合，$\lambda=58$ の日程が1個得られている．1回の試行で $\lambda=55$ を得た $\Delta\lambda$ 優先順位規則はこの種の問題にはかなり強力であると云える．さらに brache-and-bound 法による最適解の探索に必要な計算量も，これに比してかなり order が低い．

各 item の機械順序が一定であるいわゆる FLOW SHOP 型の問題に対しては，この手法はあまり有効ではない．機械順序一定という強力な制約条件が考慮されていない以上当然ではあるが，実用上注意を要する．§6 の branch-and-bound 法を用いればもちろん解は得られるが，この場合も別の問題として考えた方がよい[2]．

PERT の手法と関連していわゆる山崩し法[3][4] が用いられているが，本論文の資源順序対を指定する方法との割付方式の差違は次の点にある．この方法が時刻 t で完了する作業 ($EF_i=t$) も含めてコンフリクト作業集合を作るのに対して，山崩し法では時刻 t から実行する作業の間だけでコンフリクトを作っている．したがって山崩し法では少し遅れて到着する予定の critical な作業を待つために，現在時刻で開始できるはずの作業を遅らせるという種類の判断はできない．ただし作業分割を許す場合 (§5,2) にはこの差違はなくなる．$\Delta\lambda$ 優先順位は作業 J については LS_J 優先，すなわちトータル・フロート優先順位規則と同じである．ただし EF_J が小さい場合にこれが I に入る機会が生ずる点でこれと異なっている．

以上順序指定型の人員機械配分法についてその考え方をまとめて報告したが，計算機の上での実際計算は行なっていない．実用上の評価も含めて，今後の課題としたい．

この論文の作成に当って，種々の有効な御指導・御検討をいただいた慶応大学工学部の山内二郎教授・関根智明助教授に，深く感謝の意を表します．

参 考 文 献

1. Fisher, H. and Thompson, G.L., "Probabilistic Learning Combinations of Local Job-Shop Scheduling Rules", *Industrial Scheduling*, Pretice-Hall, 1963.
2. Broun, A.P.G. and Lomnicki, Z. A., "Some Application of the Branch-and-bound Algorithm to the Machine Scheduling Prolem," O. R. Q. Vol. 17. No. 2, 1966.
3. Kelley, Jr.J. E., "The Critical-path Method : Resources Planning and Scheduling," *Industrial Scheduling*, Prentice-Hall, 1963.

4 CRITICAL PATH-MANSCHEDULING REVISED (SHARE DISTRIBUTION # 1453 Roc 8001 (PA)