マルチプロジェクト・スケジューリング†

大前義次†† 長谷川晋†† 奥田稔††*

プロジェクト・スケジューリングでは，通常資源制約下で各作業が実行される．特に同一資源を必要とする複数プロジェクトが同時並行的に推進される場合，資源の競合が頻発し，問題が一層複雑となる．有限資源制約下でのマルチプロジェクト・スケジューリング問題に対して，数理計画法に基づく最適化算法が種々提案されているが，この種の問題が組み合わせ問題であるため，取り扱うことのできる問題の規模に限界がある．実用的方法としては，適当な優先規則を用いて実行スケジュールを与える発見的方法が提案されており，この種のものとして RAMPS や ISMPS がある．本研究は ISMPS をベースに改良を加えたものであり，資源制約下での複数プロジェクトを対象とし，管理要因の選定と，それらに適切な重み付けを行うことによって最良スケジュールを決定するものである．スケジュール結果は，プロジェクトを構成するデータの種類やスケジューリングの目的によって大きく変化する．本研究では，管理要因の重み値の組み合わせをスケジューリングの都度学習によって決める方法を提案している．また特に指定されたプロジェクトの遅延日数を短縮するための調整法についても提案している．

1. まえがき

情報システムの開発を初めとして多くの大規模プロジェクトの推進に当たって，スケジュールの計画と管理は，極めて重要な問題である．PERT や CPM のような，一般に広く用いられている方法は，資源の制約のない単一プロジェクトの実行時間や，それと費用の関係などを求めている．しかし，実際のプロジェクトの推進では，同一の資源を必要とする複数プロジェクトが同時並行的に推進され，問題は一層複雑となる．

このような資源制約下での複数プロジェクト実行において，総所要時間（プロジェクト実行時間），総納期遅れ時間，全プロジェクト完了時間などの目的関数を最小にするために，資源を各作業にどのように配分したらよいかという問題を以下に考察する．ここでプロジェクトを構成する各作業は，それぞれ所与の先行関係に従って処理され，その処理の仕事量は与えられているが，資源の制約下で同時に処理できない作業をどのようにスケジュールしたら目的達成に効果的かを決めなければならない．

有限資源の制約下での複数プロジェクトのスケジューリング問題に対して，これまで整数線形計画法で定式化する方法[1]，分枝限定法による方法[2]，Backtrack 法による方法[3]等最適化算法が試みられている．しかしこの種の問題は組み合わせ問題であるために，これら数理的計画法で扱うことのできる問題の規模には限界があり実用的とはいいがたい．そこで適当な優先規則を用いて実行可能スケジュールを求める発見的方法が，提案されている．この種の解法の代表的なものとして C.E.I.R 社が開発した RAMPS[4] や大前と金が開発した情報システム開発のための ISMPS (Information System Multi-Project Scheduling)[5] 等がある．しかし前者の RAMPS では，その核心部分であるスケジュールの実行パターンを決定する Algorithm は公開されていない．

本研究は ISMPS をベースに改良を加えたもので，PERT 型のスケジューリングではあるが，資源制約下での複数プロジェクトを対象とし，スケジュールをコントロールするための管理要因を選定し，それぞれに重み付けのための計算式を導き，それによって総合的スコアを計算し，最良スケジュールを決定することを可能とするものである．スケジュール結果は，プロジェクトを構成するデータの種類やスケジューリングの目的によって大きく変化する．したがって，本研究では，これらに対応するためにこれまでの手法のように重み値を固定せず，Algorithm をスケジュール計算 Process と最適重み値の学習 Process に分け，最適重みの組み合わせは，その都度，スケジューリングの目的に対応して学習するという新しい概念を導入した．

なお，ISMPS では，資源制約として SE，プログラマ，コーダ等各種の人的資源を考慮しているが，ここでは資源として1種類の作業人員だけを取り上げている．この意味で本研究は単一資源制約の場合といえる．

† Multi-Project Scheduling by YOSHITSUGU OHMAE, SUSUMU HASEGAWA and MINORU OKUDA (Department of Information Science, Faculty of Engineering, University of Ibaraki).

†† 茨城大学工学部情報工学科
* 現在 日本電信電話(株)

さらに本研究では，スケジューリングによって導かれた各プロジェクト計画のうち，特定プロジェクトの実行期間を繰り上げて遅延日数を短縮するための調整法についても提案している．

2. スケジューリングの基本事項

マルチプロジェクト・スケジューリングの基本事項とスケジューリングの概要について以下に述べる．

2.1 プロジェクト・スケジューリングの管理目標

本研究ではプロジェクト・スケジューリングの管理目標を次のように設定している．

（1） プロジェクト実行期間の短縮

すべてのプロジェクトをできるだけ短い期間で完成するようにする．

（2） プロジェクト別の期間要求（納期）の達成

各プロジェクトには納期が与えられている．納期遅れがでないような日程計画を組む必要がある．

（3） 資源の有効利用

利用可能な資源を遊ばせることなく，資源の有効利用をはかる．ここでは資源として作業人員を対象としている．

2.2 管理要因とその影響

管理目標を達成するための管理手段として，本研究では，管理要因を定めている．考慮すべき管理要因としては，**表1**に示すようなものが考えられる．これらの要因は，これまで多くの研究で取り上げられてきたものである．

表1にあげたもののうちで，次の四つのものを本研究では管理要因として採用する．

① 余裕日数 (Float Days)：FD
（これは TF と FF をあわせたもの）

② 最遅開始時刻 (Latest Start Time)：LS

③ 最短作業 (Shortest Activity)：SA

④ 実行済み仕事量 (Finished Work Amount)：FW

他の要因は，以下の理由で除いた．

WC：これと同等であると解釈できる FW を管理要因として選んだ．

UR, SJ：Algorithm 上で考慮に入れることができるので管理要因からは除いた．

RT：これは SA と同等であるが，SA はクリティカル・パスも考慮しているのに対して，RT は局所的なものであるからである．

SC：直接後続作業の多い作業を優先しても隘路の防止にはならないため除いた．

IR：プロジェクト別に重要度を与えることは，何を基準に順位付けるか不明確であるので除いた．

2.3 基本要素

各 Activity（作業）の仕事量を次式で表す．

仕事量(工数)＝速度別必要単位資源量＊所要時間 (2.1)

表1 管理要因とスケジュール上の影響
Table 1 Control factors and effects on schedule.

管理要因名	記号	要因と管理方法	スケジュール上の影響				関連文献
			期間短縮	隘路防止	遅延防止	資源活用	
1. 全余裕日数 (Total Float)	TF	余裕のない作業を優先	Y		Y		4), 5), 6), 7), 8), 10)
2. 自由余裕日数 (Free Float)	FF	余裕のない作業を優先	Y	Y			4), 5), 6), 7)
3. 最遅開始時刻 (Latest Start Time)	LS	この時刻が早いものを優先			Y		5), 9)
4. 作業の連続性 (Work Continuity)	WC	既に実施中の作業は中断しないよう優先		Y		Y	4), 6), 7)
5. 実行済み仕事量 (Finished Work)	FW	計算時点で，既に実行された仕事量の多い作業を優先		Y			8), 10)
6. 遊休資源 (Unemployed Resource)	UR	未使用の資源の割り当て優先				Y	4), 6), 7)
7. 計画作業数 (Scheduled Job)	SJ	できるだけ多くの作業の開始を優先		Y			4), 6), 7)
8. 残り作業時間 (Remaining Time)	RT	残り作業時間の短いものを優先	Y	Y			5)
9. 最短作業 (Shortest Activity)	SA	計算時点で作業時間が短く，クリティカル・パスの最短のものを優先	Y	Y	Y		8), 10)
10. 後続作業数 (Successors)	SC	後続作業数の多い作業を優先		Y			4), 5), 6), 7)
11. 重要度 (Important Rate)	IR	Project に重要度をつけて重要度の高い方を優先			Y		5)

(注) Y：関連文献で取り上げているもの

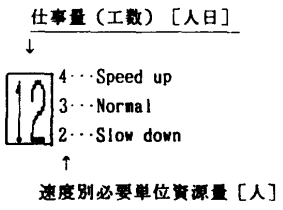


図 1 作業の表記例
Fig. 1 Explanatory example on activity.

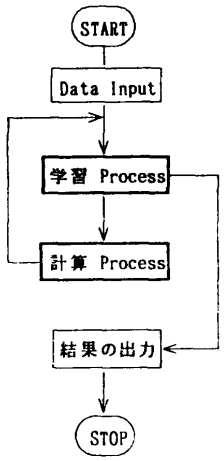


図 2 スケジューリングの処理フロー
Fig. 2 Flow of scheduling procedure.

本式は工数の式として広く実用に供されている．しかし，人数を増やせばその分だけ日数を減らせるといえない場合もあるが，本研究では本式の関係を仮定する．また，各作業を三段階の速度 (Speed up, Normal, Slow down) のいずれかによって実施する．競合する人員の調整は作業速度を変化させることによって行う (図 1)．

本研究の前提条件として，各プロジェクトごとの開始日は与件として設定されており，遅れることはあっても早めることはできないものとする．

2.4 スケジューリング処理フロー

スケジューリング処理フローの概要を図 2 に示す．ここで学習 Process と計算 Process が特に重要な部分である．

3. 計算 Process の Algorithm

計算 Process の Algorithm の概要を図 3 に示す．これら Algorithm の中で，②各作業のスコア計算，③各作業の仮速度の決定，④Schedule の実行パターン決定の三つが核心ルーチンであるので，以下ではこれについて重点的に述べる．

本方法では，決められた時点でスケジュールに必要な各要素を PERT によって計算することを基本としている．したがってはじめに計算時点から述べる．

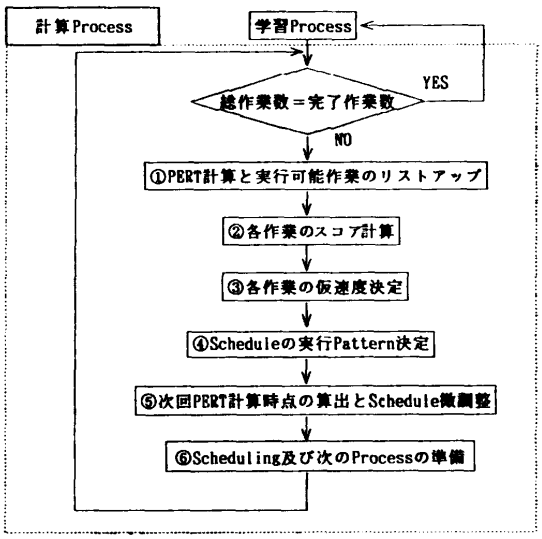


図 3 計算 Process の Algorithm
Fig. 3 Algorithm of computation process.

3.1 スケジュールの計算時点

スケジュール計算時点として次の時点が考えられる．

① 最初の計算時点
　スケジュールの開始時点（0時間）．

② プロジェクト内の作業が完了した時点
　プロジェクトを構成するある作業が完了したとき，その作業の後続作業が次スケジュールの対象となる．

③ 新たにプロジェクトが開始される時点
　すべてのプロジェクトが同時刻に開始できるわけではない．したがって新たに開始されるプロジェクトがあるときは，それを計算時点とする．

④ プロジェクトに変化が生じた時点
　実行中のプロジェクトに変化（期間要求等の変更）が発生したとき，また特急プロジェクトの割り込みが発生したときなどは，その時点を計算時点とする．

⑤ 定期的計算
　決められた期間単位（一週間単位等）で計算する．

本研究では，⑤の定期的計算は行わない．その理由は，①から④はプロジェクトの作業と資源量に何らかの変化が生じたときであり，変化の生じてない時点に計算時点をもうけてもスケジュールに影響を与えないからである．

3.2 スケジューリング実施例

スケジューリングの対象となる三つのプロジェクトA，B，Cを**図4**に示す．これらプロジェクトA，B，CをNormal Speedで，資源制約がない場合でスケジューリングすると完了日は，それぞれ16日，11日，14日となる．**図5**は，その作業人員の山積み表であるが，7日目に25人の人員が必要となりピークを構成している．また，資源利用率（以下利用率という）も58.5%と低い．

表2は図4の例に対して資源制約を25[人]→15[人]とし，本研究の計算Algorithmを適用したときの7日目のPERT計算表である．ただし，各プロジェクトA，B，Cの納期(PR)は，16日，11日，14日としてある．

表2のPCが−1になっているものは，その作業が完了したことを示しており，したがってこの時点ではANの1, 3, 9, 10, 11, 13, 16, 17, 18が終了している．

ここで，実行可能な作業は，PC＝0のものであるから，これらをリストアップするとANの2, 4, 6, 12, 14, 19, 20, 21となる．

3.3 各作業のスコア計算

スコア計算は，基本的に以下の三つのStepからなる．

Step 1：管理要因別ベーススコアの計算．

Step 2：管理要因別スコアの計算．

Step 3：作業別スコアの計算．

四つの管理要因LS, FD, FW, SAに対して与える重みは，学習Processで最適値を見出すことにしているが，以下では説明の便宜上，重みを次のように与える．

LSW＝30, FDW＝10, FWW＝20, SAW＝40

ここで各管理要因の末尾のWは重みを表す．また，重みの総和は次のように100とする．

$$LSW+FDW+FWW+SAW=100 \tag{3.1}$$

Step 1：管理要因別ベーススコアの計算．

各管理要因のベーススコアの計算は，次式による．

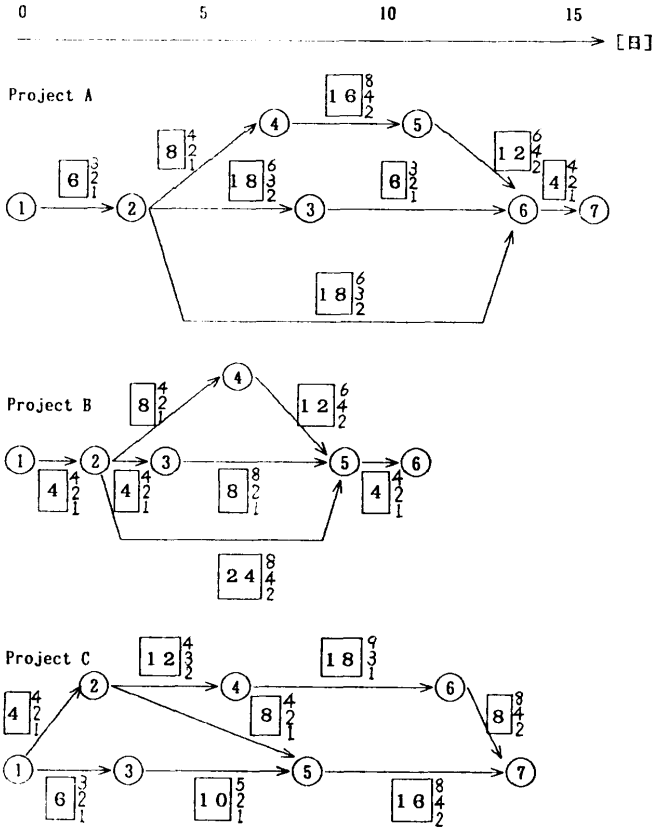


図4 3-Projectの場合の例
Fig. 4 Example of 3-projects case.

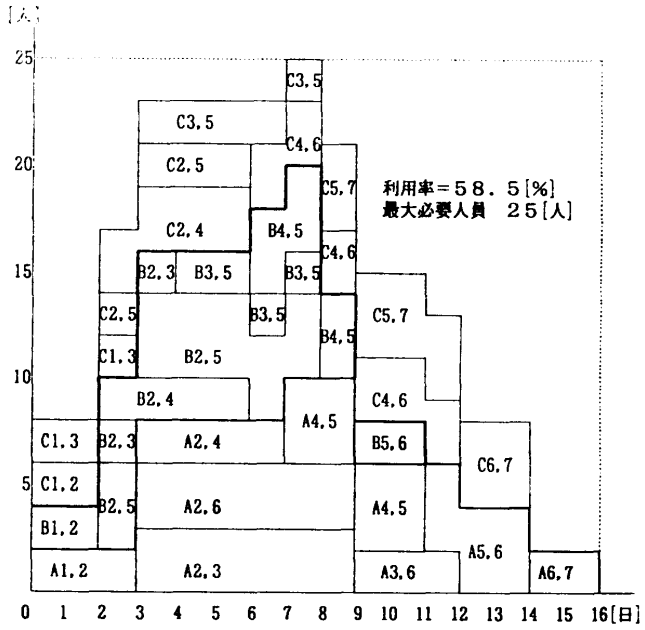


図5 図4のNormal Speedでの人員山積み表
Fig. 5 Load chart at normal speed of Fig. 4.

表 2 本研究の計算 Algorithm を適用した7日目時点の PERT 計算値
Table 2 PERT calculation of computation process at 7th day.

AN	PN	Ni	Nj	PC	D	W	FW	LS	TF	FF	PF	PR
1	1	1	2	−1	0	6	6	*	*	*	*	16
2	1	2	3	0	6	18	0	7	0	0	18	16
3	1	2	4	−1	0	8	8	*	*	*	*	16
4	1	2	6	0	6	18	0	10	3	3	18	16
5	1	3	6	1	3	6	0	13	0	0	18	16
6	1	4	5	0	4	16	0	9	2	0	18	16
7	1	5	6	1	3	12	0	13	2	2	18	16
8	1	6	7	3	2	4	0	16	0	0	18	16
9	2	1	2	−1	0	4	4	*	*	*	*	11
10	2	2	3	−1	0	4	4	*	*	*	*	11
11	2	2	4	−1	0	8	8	*	*	*	*	11
12	2	2	5	0	1	24	22	7	0	0	10	11
13	2	3	5	−1	0	8	8	*	*	*	*	11
14	2	4	5	0	1	12	10	7	0	0	10	11
15	2	5	6	2	2	4	0	8	0	0	10	11
16	3	1	2	−1	0	4	4	*	*	*	*	14
17	3	1	3	−1	0	6	6	*	*	*	*	14
18	3	2	4	−1	0	12	12	*	*	*	*	14
19	3	2	5	0	2	8	5	9	2	0	15	14
20	3	3	5	0	2	10	7	9	2	0	15	14
21	3	4	6	0	6	18	0	7	0	0	15	14
22	3	5	7	2	4	16	0	11	2	2	15	14
23	3	6	7	1	2	8	0	13	0	0	15	14

(注) *：既に完了した Activity
AN：Activity Number　PN：Project Number
Ni：Node i (出発 Node)　Nj：Node j (到着 Node)
PC：先行作業数　D：所要日数
W：仕事量　FW：実行済み仕事量
LS：最遅開始時刻　TF：全余裕日数
FF：自由余裕日数　PF：Project の完了予定日
PR：Project の納期

① LS のベーススコア：LSB

LSB＝LS 値＋計算対象のプロジェクトの納期
　　−計算対象のプロジェクトの完了予定日
　　＝LS 値＋PR 値−PF 値　(3.2)

LSB については，この値が小さいほど優先される．

② FD のベーススコア：FDB

FDB＝(TF 値＋FF 値)＊PSV　(3.3)

ここで PSV (Project Size Value) とは，全プロジェクトの中で最も大きいプロジェクトの完了予定日を，計算対象のプロジェクトの完了予定日で割ったものであり，管理要因が，プロジェクトの大，小にかかわらず，同じ条件で比較できるようにするためのものである．PSV は次式で与えられる．

表 3 各作業のベーススコア
Table 3 Base scores for each activity.

AN	LSB	FDB	FWB	SAB
2	5	0.0	0.000	17
4	8	6.0	0.000	17
6	7	2.0	0.000	15
12	8	0.0	0.917	4
14	8	0.0	0.833	4
19	8	2.4	0.625	10
20	8	2.4	0.700	10
21	6	0.0	0.000	14

$$PSV=\frac{\text{Max (PF)}}{\text{計算対象のプロジェクトの PF 値}} \quad (3.4)$$

FDB については，その値が小さいほど優先される．

③ FW のベーススコア：FWB

$$FWD=\frac{\text{実行済み仕事量}}{\text{仕事量}}=\frac{\text{FW 値}}{\text{W 値}} \quad (3.5)$$

FWB については，この値が大きいほど優先される．

④ SA のベーススコア：SAB

SAB＝作業の所要日数
　　＋計算対象のプロジェクトの完了予定日
　　−計算時点＝D値＋PF 値−PT　(3.6)

SAB については，この値が小さいほど優先される．

以上の①〜④から，表2の7日目に実行開始可能な作業のすべてに対して各要因別ベーススコアを計算すると**表3**のようになる．

Step 2：管理要因別スコアの計算．

管理要因別スコアの計算は次式による．

管理要因のスコア＝(各要因の重み値)＊(各要因のベーススコアによる順位得点)　(3.7)

ここで各要因のベーススコアによる順位得点という考え方を導入する．その理由は次による．

ベーススコアによる順位得点によらないで，直接各要因のベーススコアを用いると各要因のベーススコアは基準化されていないため，重み値を掛ける意味が失われるからである．そこで各要因のベーススコアは重み値を掛ける前に，その値を基準化する．基準化の方法として各要因ごとのベーススコアを順位付け，得点化することによって行う．この方法は本質的にゴルフトーナメントの賞金の分け方と同様である．この方法を用いれば，重みを掛ける前の値の範囲を1から N (N は計算時点で実行可能な作業数) に統一でき，その値の合計（得点合計）も，

$$\text{合計}=\sum_{K=1}^{N} K \quad (K \text{ は } 1,\cdots,N \text{ の整数}) \qquad (3.8)$$

に統一できる．この順位得点は次式による．

$$\text{順位得点}=\frac{\sum_{K=S}^{F} K}{\text{SAME}} \qquad (3.9)$$

ただし，ここで

$F=N-\text{UPER}$

$S=F-\text{SAME}+1$

N：計算時点で実行可能な作業数

K：$S, S+1,\cdots,F$ の整数

UPER：計算対象のベーススコアより優先されるものの数．

SAME：計算対象のスコアとその値が等しいものの数（ただし，計算対象自身も含む）．

表3の LSB についてこれを適用すると**表4**のようになる．さらに表3のすべての要因に適用すると，**表5**のようになる．表5の各管理要因の末尾のPは順位得点を示す．

さて，管理要因別スコアは，この順位得点に重みを掛ければ良いので**表6**のようになる．

ここで各管理要因の末尾のSはスコアを示す．

Step 3：作業別スコアの計算

作業別スコアの値は管理要因別スコアの総和であり，次式で表される．

$$\text{作業別スコア}=\text{LSS}+\text{FDS}+\text{FWS}+\text{SAS} \qquad (3.10)$$

これを表6の全作業に適用すると**表7**のようになる．

3.4 各作業の仮速度決定

3.3 節で計算された作業別スコアに基づいてスケジューリングのための作業別仮速度を決める．なおここで得られた速度値（Speed up：3, Normal：2, Slow down：1）は，Schedule の実行パターン決定時に制限資源にしたがって変化するので仮速度と呼ぶ．仮速度決定においては，作業ごとに速度の適切な割り当ての基準となる振り分け値が必要である．本研究ではスケジュール計算時点で実行可能な作業の中で，作業別スコアの最大のものを選び，そのスコア値の 1/3, 2/3 点を振り分け値とする．次式では，最大のスコア値を MS (Maximum Score) とする．

① 速度 3：2/3＊MS≦作業のスコア

② 速度 2：1/3＊MS＜作業のスコア＜2/3＊MS

③ 速度 1：1/3＊MS≧作業のスコア

振り分け値によって算出された速度値はその値が大きい作業ほどスケジューリングで優先される．

このように決定された仮速度を**表8**に示す．

表4 LSB の順位得点
Table 4 Ranking points of LSB.

AN	LSB	UPER	SAME	F	S	順位得点
2	5	0	1	8	8	8
4	8	3	5	5	1	3
6	7	2	1	6	6	6
12	8	3	5	5	1	3
14	8	3	5	5	1	3
19	8	3	5	5	1	3
20	8	3	5	5	1	3
21	6	1	1	7	7	7

表5 各要因の順位得点
Table 5 Ranking points for each factor.

AN	LSP	FDP	FWP	SAP
2	8.0	6.5	2.5	1.5
4	3.0	1.0	2.5	1.5
6	6.0	4.0	2.5	3.0
12	3.0	6.5	8.0	7.5
14	3.0	6.5	7.0	7.5
19	3.0	2.5	5.0	5.5
20	3.0	2.5	6.0	5.5
21	7.0	6.5	2.5	4.0

表6 各管理要因のスコア
Table 6 Scores for each factor.

AN	LSS	FDS	FWS	SAS
2	240	65	50	60
4	90	10	50	60
6	180	40	50	120
12	90	65	160	300
14	90	65	140	300
19	90	25	100	220
20	90	25	120	220
21	210	65	50	160

表7 作業別スコア
Table 7 Scores for each activity.

AN	作業別スコア
2	415
4	210
6	390
12	615
14	595
19	435
20	455
21	485

表 8 各作業の仮速度
Table 8 Temporary speeds for each activity.

AN	作業別スコア	仮速度値
2	415	3
4	210	2
6	390	2
12	615	3
14	595	3
19	435	3
20	455	3
21	485	3

3.5 Schedule の実行パターン決定

（1） Schedule の実行パターン決定の Algorithm

Schedule の実行パターン決定のための Algorithm は，次の**図 6** で表されるようにA～Dの四つの Schedule 算出パターンから構成される．これらは資源を最大限利用するために必要となるものである．

ここで，作業資源（各作業の利用資源量の総和のこと．）が仮速度の段階で制限資源量と一致する場合は，仮速度をそのまま作業別速度値として次のルーチン（次回 PERT 計算時点の算出と Schedule 微調整）へ移る．そうでない場合は，Schedule 算出パターンAから順に行い作業資源と制限資源量が一致したところで次のルーチンへ移る．ここで微調整というのは，ある作業の速度値をさらに下げても日程に影響を与えない場合，この浮かせた資源を他の作業にまわし，有効に使うための調整である．

最後の算出パターンDへ進み，なお作業資源と制限資源量が，一致しない場合は制限資源量の範囲内で最大の利用資源量を満たす速度値を作業別速度値としてこのルーチンを抜ける．

（2） Schedule 算出パターンの構成

Schedule 算出パターンは，優先順位と資源調整の二つの Algorithm からなる．さらに，それぞれに二つのパターンがあるので，これらを組み合わせると図6に示したようにSchedule算出パターンはA～Dの四つになる．これらを以下に示す．

① Schedule 算出パターンA

優先順位：パターン1，資源調整：パターン1

② Schedule 算出パターンB

優先順位：パターン2，資源調整：パターン1

③ Schedule 算出パターンC

優先順位：パターン1，資源調整：パターン2

④ Schedule 算出パターンD

優先順位：パターン2，資源調整：パターン2

ここで，優先順位パターン1は，各作業は仮速度の大きい順に優先される．同じ速度間では作業別スコア値の大きいものが優先される．優先順位パターン2では，パターン1と同様の仮速度の大きい順に優先されるが，同じ速度間では，現在の速度値（仮速度値）と一つ下の速度値の必要単位資源量の差が大きいものが優先される．

資源調整 Algorithm は，作業資源と制限資源量の大小関係からさらに次の二つに分かれている．

1) 資源余裕のある場合（作業資源＜制限資源量）

この場合は，余裕資源をもっと使うように作業別速度値を上げるルーチンである．作業別速度値を優先順位の高い方から上げて，その分の利用資源を加え，新しく算出された作業資源と制限資源量との比較をする．この際，制限資源量の範囲内で最大資源利用の作業別速度値パターンを Save する．

2) 資源不足の場合（作業資源＞制限資源量）

この場合は，作業別速度値を下げて作業資源を減らすルーチンである．作業別速度値を優先順位の低い方から下げてその分の資源量を引き，作業資源が制限資源量の範囲内に入るまでこれを繰り

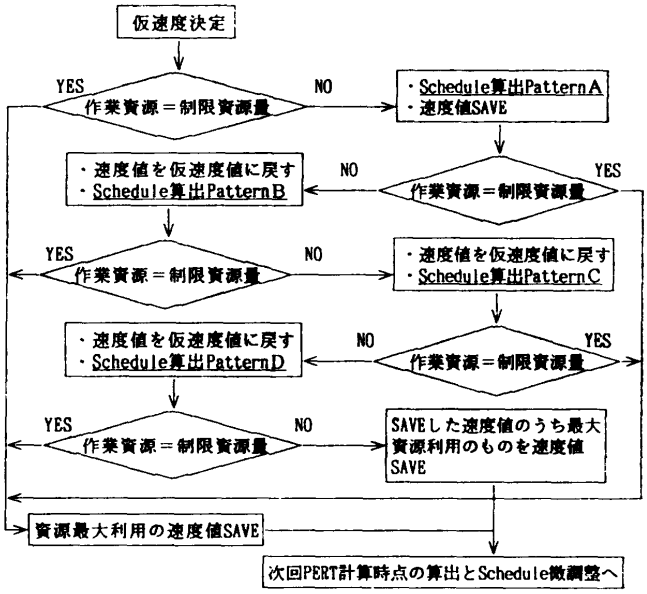


図 6 Schedule の実行 Pattern 決定 Algorithm
Fig. 6 Determination algorithm of execution pattern of schedule.

表 9 Schedule の実行 Pattern 決定
Table 9 Determination of execution pattern of schedule.

スケジュール対象作業の状態				資源余裕ルーチン適用					
AN	スコア	3	2	1	仮速度	資源	速度	資源	速度	資源
12	615	8	4	2	3	8	2	4	2	4
14	595	6	4	2	3	6	2	4	3	6
21	485	9	3	1	3	9	2	3	2	3
20	455	5	2	1	3	5	2	2	2	2
19	435	4	2	1	3	4	2	2	2	2
2	415	6	3	2	3	6	2	3	2	3
6	390	8	4	2	2	4	1	2	1	2
4	210	6	3	2	2	3	1	2	2	3
総資源量						45		22		25

資源不足ルーチン適用

返す．その結果得られた最大資源利用の作業別速度値パターンを Save する．

ここで，優先順位の高いまたは低い方から速度値を上げ下げする部分があるが，この部分には二つの資源調整のパターンがある．

① 資源調整パターン1

優先順位の高い方（低い方）から順に速度値を+1(−1)し，作業資源を算出する．

② 資源調整パターン2

優先順位の高い方（低い方）から順に速度値を制限資源量の範囲内で必要かつ可能なら上げられる（下げられる）限り上げる(下げる)．

(3) Schedule の実行パターン決定ルーチンの適用例

以上の方法を表8に適用する．この場合仮速度での作業資源を算出すると表9のように 45 [人] となる．制限資源を 25 [人] とすると，これからまず Schedule 算出パターンAの優先順位パターン1，資源調整の資源不足ルーチンへ入る．

この例では Schedule 算出パターンAの資源不足ルーチン実行時で作業資源が 22 [人] となり，制限資源量 25 [人] を割り込むので，再び資源余裕ルーチンを適用する必要がある．実施した結果制限資源量と一致したのでこの値を最大資源利用の速度値として Save し，Schedule の実行パターン決定のルーチンを抜ける．

4. 最適重み値の学習 Process

4.1 スケジューリングの実施例

図5と同じ例に対して制限資源が 15 [人] という厳しい制約条件の下で本研究の方法を適用した結果は図7のとおりである．ただし，管理要因の重み値は，3.2節の例の場合と同じとする．

プロジェクト A，B，C の完了日は，それぞれ 17日，9日，14日となっており，これは資源制約なしのスケジュールに比べプロジェクトAがわずかに1日遅れただけで，プロジェクトBに関しては，2日も早く終わっている．

プロジェクトAは，1日遅れたが，厳しい制約条件を考慮すれば，スケジュールが大幅に改善されたと判断される．

また，資源利用率も 96.0% となり，大幅に改善された山崩しといえる．

4.2 スケジューリングの評価尺度

スケジューリング実施に当たり評価尺度が必要である．本研究では，以下に示す三つの評価尺度と，一つの参考尺度を用いる．評価尺度として，

① 総所要時間 (Total Duration): TD

すべてのプロジェクトの所要時間（期間）の総和である．次式による．

$$TD=\sum_{i=1}^{h}(Finish_i-Start_i) \qquad (4.1)$$

ここで，i: 1 から h までのプロジェクト番号．

$Finish_i$: プロジェクト i の完了した時刻

$Start_i$: プロジェクト i の実行開始時刻

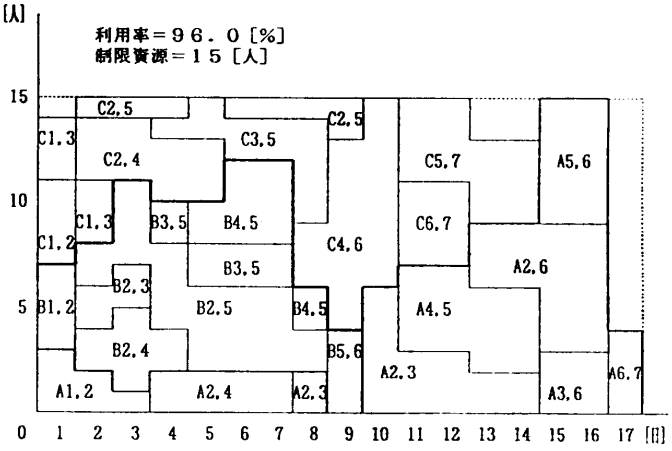


図 7 サンプル・スケジュールの山積み表
Fig. 7 Load chart of sample schedule.

TD は，小さいほど良いスケジュールといえる．

② 総納期遅れ時間 (Total Penalty Delay)：TPD

すべてのプロジェクトの納期遅れ時間の総和である．次式による．

$$TPD=\sum_{i=1}^{h} \text{Penalty Delay}_i \qquad (4.2)$$

ただし，Penalty Delay$_i$ は，次式による．

Penalty Delay$_i$：

if $PR_i \geqq Finish_i$ then Penalty Delay$_i=0$

else Penalty Delay$_i=Finish_i-PR_i$

ここで，PR_i はプロジェクト i の納期である．

TPD は小さいほど良いスケジュールといえる．

③ 全プロジェクト完了時間 (All Project Finish Time)：APFT

すべてのプロジェクトが，完了した時間である．次式による．

$$APFT=\underset{i\in h}{Max}(Finish_i) \qquad (4.3)$$

APFT は小さいほど良いスケジュールといえる．

④ 資源利用率 (Resource Utilization)：RU

すべてのプロジェクト実行のための資源利用率．

$$RU=\frac{TWA}{MAR*APFT}*100 \qquad (4.4)$$

ここで，TWA：総作業仕事量(Total Work Amount)
実際に作業した延べ仕事量

MAR：最大配置資源量(Maximum Assigned Resources) プロジェクト遂行期間中の最大配置資源量

この値が大きいほど，スケジュールにおいて，資源利用が平滑されていることを示す．しかし，この値は，余剰資源を強引に作業に割り当てれば上げることができ，他の目的に照らした場合大きいほど良いとはいえないのでここでは，これを参考尺度としている．

4.3 学習 Process

本研究では，管理要因の最適重み値の組み合わせは，次の学習 Process によって決定することを特徴としている．

（1） 学習する重み値の組み合わせ

次の二つのルールで決まる管理要因の重み値を学習によって決める．

① それぞれの管理要因の重み値のとり得る値は，10刻みで 0～100 とする．

② 管理要因の重み値の合計は 100 とする．このような重み値の組み合わせは 286 通りある．これらすべてを用いて最適重み値の組み合わせを学習する．さらに精度を上げるため刻みを小さくしたい場合でも，以上の学習によって得られた重み値の近傍の値について調べれば良く，実用時間で計算できる．

（2） 学習方法

4.2 節でスケジュールの結果を評価するための評価尺度を三つあげたが，これらは，管理目標としての目的関数でもある．本研究では，重み値の組み合わせの学習にあたり，この目的関数を使い，重み値の各々の組み合わせについてコンピュータ・シミュレーションによって評価尺度別の最適重み値の組み合わせを求める．

4.4 シミュレーション・データ

シミュレーションのための入力データは，資源制約なしの Normal Speed でスケジューリングした結果得られる山積み表から，(a)前半 Peak 型，(b)中央 Peak 型，(c)Flat 型の三つに大別した．これを**図 8** に示す．

表 10 に示すように，A～P の 16 個の単独プロジェクトを用意し，これらをそれぞれ三つずつ組み合わせて，EX 1～EX 6 のシミュレーション・データを作成した．

4.5 シミュレーション結果と考察

表 11 は，評価尺度別の最適重み値の組み合わせと

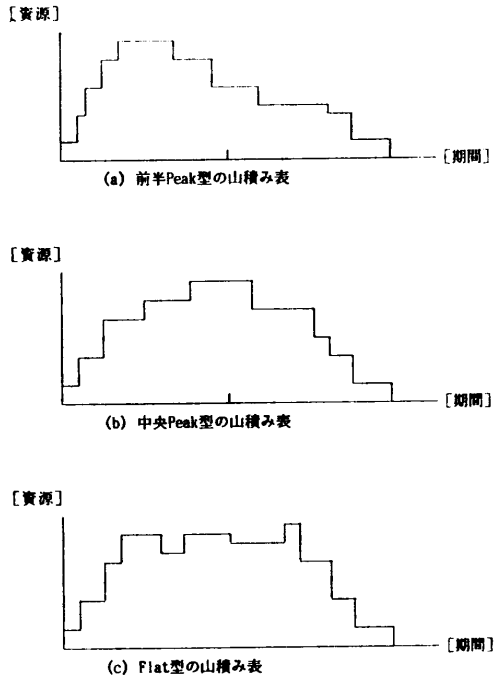


図 8 人員山積み表
Fig. 8 Load profile.

目的関数の最良値を示したもので，重み値の並びは (LSW, FDW, FWW, SAW) である．

表 12 は資源制約なしの Normal Speed でのスケジュール結果と，シミュレーション結果の最良値と最悪値を示したものであるが，Normal Speed 時の総納期遅れ時間が 0 となっているのは，資源制約なしの Normal Speed でスケジュールしたときの各プロジェクトの完了日をプロジェクトの納期としたからである．

表 12 から，資源制約なしの Normal Speed の下でスケジューリングしたものと，本方法を適用したときの最良値を比較し，評価をすると，厳しい資源制約下にあることを考えれば，本方法のスケジュール結果は，かなり良いものであることがわかる．特に前半 Peak 型 (EX 1, EX 2) においては，優れたスケジュール結果が示されているが，これは，Flat 型や中央 Peak 型

表 10 シミュレーション・データ
Table 10 Simulation data.

例	データの型	Peak 時の資源量〔人〕	制限資源量〔人〕	組み合わされるプロジェクト名
EX 1	前半 Peak 型	152 (42.97%) *1	80	Project A (0～183) *2 Project B (0～105) Project C (0～ 88)
EX 2	前半 Peak 型	85 (55.10%)	50	Project D (0～ 86) Project E (0～ 64) Project F (0～ 32)
EX 3	中央 Peak 型	151 (43.14%)	80	Project K (0～183) Project L (48～153) Project M (40～128)
EX 4	中央 Peak 型	47 (38.82%)	15	Project N (0～135) Project O (40～148) Project P (70～130)
EX 5	Flat 型	52 (60.78%)	30	Project C (0～ 88) Project D (55～141) Project G (105～192)
EX 6	Flat 型	52 (61.02%)	30	Project H (0～ 44) Project I (10～ 47) Project J (30～ 74)

(注) *1: ( )の中の数字は，資源制約なし，Normal Speed での利用率
*2: ( )の中の数字は，(プロジェクトの開始時刻～プロジェクトの納期)

表 11 評価尺度別最適重み値の組み合わせ
Table 11 Optimum weights for each evaluation measure.

例		総所要時間	総納期遅れ時間	全プロジェクト完了時間
EX 1 (前半 Peak 型)	最 良 値	375〔日〕	13〔日〕	187〔日〕
	最適重みの組み合わせ	(0, 30, 10, 60)	(0, 30, 10, 60)	(0, 40, 60, 0)
EX 2 (前半 Peak 型)	最 良 値	180〔日〕	8〔日〕	88〔日〕
	最適重みの組み合わせ	(0, 30, 20, 50) (0, 30, 0, 70)	(30, 20, 0, 50) (10, 30, 30, 30) (0, 40, 20, 40) ほか五つ	(0, 60, 40, 0) (0, 40, 60, 0) (0, 30, 70, 0)
EX 3 (中央 Peak 型)	最 良 値	389〔日〕	25〔日〕	179〔日〕
	最適重みの組み合わせ	(0, 30, 0, 70)	(0, 50, 0, 50)	(0, 50, 50, 0)
EX 4 (中央 Peak 型)	最 良 値	362〔日〕	69〔日〕	199〔日〕
	最適重みの組み合わせ	(0, 40, 10, 50)	(0, 40, 10, 50)	(20, 70, 10, 0) (30, 60, 10, 0)
EX 5 (Flat 型)	最 良 値	281〔日〕	33〔日〕	212〔日〕
	最適重みの組み合わせ	(30, 0, 30, 40)	(20, 20, 20, 40) (10, 40, 10, 40)	(40, 20, 10, 30)
EX 6 (Flat 型)	最 良 値	132〔日〕	20〔日〕	88〔日〕
	最適重みの組み合わせ	(20, 30, 0, 50)	(50, 10, 0, 40)	(50, 10, 0, 40)

に比べて，早い時間にすべてのプロジェクトが開始されるためであることが推測される．

次に，表 11 の評価尺度別の最適重み値の組み合わせを見ると，その最適な組み合わせは，本質的には評価尺度ごと，データの型等で変化しているが，一般的傾向として評価尺度の総所要時間に対しては主として FDW, SAW が，総納期遅れ時間に対しては主として LSW, FDW, SAW が，また全プロジェクト完了時間に対しては主として FDW, FWW が大きい重み値となっていることがうかがえる．しかし，実質的な最適重み値は個々のケースによって変化している．このことは，本研究で提案した目的別，データ別に管理要因の最適重み値の組み合わせを学習することの必要性を裏付けるものであり，また，表 12 の最良値と最悪値にかなりの差があることからも重みの学習の効果は大きいといえる．

表 12 シミュレーション結果の比較
Table 12 Simulation result.

例		最大配置資源量〔人〕	総所要時間〔日〕	総納期遅れ時間〔日〕	全プロジェクト完了時間〔日〕
EX 1	Normal	152	376	0	183
	最良値	80	375	13	187
	最悪値	80	466	90	213
EX 2	Normal	85	182	0	86
	最良値	50	180	8	88
	最悪値	50	215	33	104
EX 3	Normal	151	376	0	183
	最良値	80	389	25	179
	最悪値	80	469	60	230
EX 4	Normal	47	303	0	130
	最良値	15	362	69	199
	最悪値	15	477	180	232
EX 5	Normal	52	261	0	192
	最良値	30	281	33	212
	最悪値	30	440	82	232
EX 6	Normal	52	125	0	74
	最良値	30	132	20	88
	最悪値	30	189	39	97

5. 指定プロジェクトの納期遅延調整方法

以上の方法の適用時に，指定プロジェクトの納期調整をしたいという問題が生ずることが考えられる．

例えば，同時期にA，B，Cという三つのプロジェクトをこなさなければならないとき，AとCのプロジェクトはそれぞれ納期に多少遅れてもかまわないが，Bのプロジェクトは，その納期に絶対に間に合わせなくてはならないといった場合である．このように特定のプロジェクトのみに特別な要求がなされたときの対応方法について以下述べる．

この場合次のような二つの遅延調整目標をおく．重要なものから示すと，

① 指定プロジェクトの Penalty Delay を 0 にする．
② Total Penalty Delay を最小におさえる．

①は，指定プロジェクトを納期に納めることを意味するが，②は，指定プロジェクトを納期に納めようとすれば，必然的に他のプロジェクトにそのしわ寄せが及ぶことが予想されるため，指定プロジェクトを納期に納める上で，その他のプロジェクトの遅れも最小におさえようというねらいである．

指定プロジェクトを納期に納めることは，スケジューリングの際にそのプロジェクトを優先させるということである．指定プロジェクトに属する作業のスコアを大きくすれば，相対的に優先順位は上がる．そこで作業別スコアに，以下のように優先順位修正係数(以下係数という）を掛ける方法によって指定プロジェクトに属する作業のスコアを他のものより，相対的に大きくする．

① 指定プロジェクトに属する作業の作業別スコアに1以上の係数を掛ける．
② 指定プロジェクト以外に属する作業については，その係数を1に固定する．

この方法によって，指定プロジェクトを納期に納めることが可能になる．この場合，指定プロジェクトに掛ける係数は，そのプロジェクトの終了日が納期と同じか，1～2日早くなるものの中で，全体の納期遅れが最小のものを選べば良い．

6. むすび

プロジェクト・スケジューリングに影響を及ぼす要因は多い．プロジェクトを構成する各種データ，ネットワーク構成，作業別仕事量と速度別必要単位資源量など．さらにスケジューリングの目的，すなわち所要時間の短縮，納期遅れの防止などによって結果は大きく変化する．

本研究では，最良のスケジューリングを行うために

管理要因を選定し，それらの重みの組み合わせを学習することが効果的であることを明らかにした．また，スケジュール実施後特に指定されたプロジェクトについて遅延日数を短縮するための調整方法についても示した．

本研究によって得られた方法は，複数プロジェクトのスケジューリングに際して実用的手段を提供するものと考えるものである．

参 考 文 献

1) 鍋島一郎：スケジューリング理論，pp. 232-250，司巧社 (1974).
2) Moder, J. J., Phillips, R. and Davis, E. W.: *Project Management with CPM, PERT, and Precedence Diagramming*, pp. 228-236, Van Nostrand Reinhold Company Inc. (1983).
3) Nabesima, I.: A Backtrack Programming Algorithm and Reliable Heuristic Programs for General Scheduling Problem, *Rep. Univ. Elect.-Comm.*, Vol. 24, No. 1, pp. 23-36 (1973).
4) Moshman, J., Johnson, J. and Larsen, M.: Ramps—A Technique for Resource Allocation and Multi-Project Scheduling, *Proceedings—Spring Joint Computer Conference*, pp. 17-27 (1963).
5) 大前義次，金 炳奎：情報システムに対するMulti-Project Schedulingについて，利用者指向の情報システムシンポジューム，情報処理学会，pp. 141-150 (June 1988).
6) 刀根 薫監修：PERT 講座 I～IV，東洋経済新報社 (1967).
7) 関根智明：*PERT. CPM*, 日科技連 (1965).
8) Kurtulus, I. S. and Davis, E. W.: Multi-Project Scheduling: Categorization of Heuristic Rules Performance, *Manage. Sci.*, Vol. 28, No. 2, pp. 161-172 (1982).
9) Talbot, F. B.: Resource-Constrained Project Scheduling with Time-Resource Tradeoffs: The Nonpreemptive Case, *Manage. Sci.*, Vol. 28, No. 10, pp. 1197-1210 (1985).
10) Kurtulus, I. S. and Narula, S. C.: Multi-Project Scheduling: Analysis of Project Performance, *IIE Trans.*, Vol. 17, No. 1, pp. 58-66 (1985).

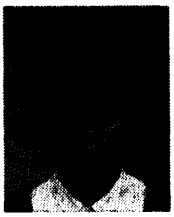
(平成2年3月28日受付)
(平成2年6月 4 日採録)

大前 義次（正会員）

1952年日本大学工学部卒業．1968年京都大学工学博士．1983年以来茨城大学工学部情報工学科教授．教育・研究分野は情報システム科学とコンピュータ・ネットワークの解析等．著書は，「応用待ち行列理論」（日科技連），「情報ネットワークの構築と運用」（オーム社）等．電子情報通信学会，OR 学会各会員．

長谷川 晋（学生会員）

1967年生．1990年茨城大学工学部情報工学科卒業．現在同大学院修士課程在学中．



奥田 稔

1967年生．1990年茨城大学工学部情報工学科卒業．同年 NTT 入社．