1　スケジューリングとは

スケジューリング（scheduling）:
稀少資源（resource）を諸活動（activity）へ（時間軸を考慮して）割り振るための方法

応用: 工場内での生産計画，計算機におけるジョブのコントロール，プロジェクトの遂行手順の決定など，様々

2　資源と活動に関する記号

資源：機械（machine），プロセッサ（processor），人員，材料，資金（予算）などを抽象化

機械の総数： m
各機械： $M_1, M_2, \cdots, M_i, \cdots, M_m$ **もしくは** $1, 2, \cdots, i, \cdots, m$ **．**

活動：ジョブ，仕事 （job）もしくはタスク（task）

ジョブの総数： n
各ジョブ： $J_1, J_2, \cdots, J_j, \cdots, J_n$ **もしくは** $1, 2, \cdots, j, \cdots, n$

オペレーション，作業（operation）：ジョブを複数の小ジョブに分解したもの
ジョブ J_j **が** m_j **個のオペレーションから構成されているとき，各オペレーションを** $O_{1j}, O_{2j}, \cdots, O_{m_j j}$ **で表す．**

3　機械スケジューリングと資源制約つきスケジューリング

機械スケジューリング問題（machine scheduling problem）：**各資源（機械）の時間軸の一部を（1 つの機械だけで）占有することによって処理される**

資源制約つきスケジューリング問題（resource constrained scheduling problem）：**資源（機械）の時間軸の一部を使用する**

⇒ **機械スケジューリング問題は，資源制約つきスケジューリング問題の特殊形**

4　単一機械と複数機械

- **単一機械（single machine）：機械が 1 台の場合**

- **複数機械**
  - **並列機械（parallel machine）：機械が複数あり，各ジョブが1 つのオペレーションから構成されており，かつジョブが任意の機械に割り振ることができる**
  - **直列機械（serial machine）：ジョブが複数のオペレーションから構成されており，かつ各ジョブに含まれるすべてのオペレーションが処理されるとジョブの処理が完了**

5　並列機械の分類

ジョブ J_j の処理時間（processing time）を p_j

- **同一並列機械（identical parallel machine）：すべての機械の速度が同じ場合**
- **一様並列機械（uniform parallel machine）：機械によって速度が決まっている場合**
  機械 M_i の速度を s_i
  ジョブ J_j の機械 M_i 上での処理時間 p_{ij} は p_j/s_i
- **無相関並列機械（unrelated parallel machine）：**
  ジョブによって機械の速度が異なる場合も考えられる．
  ジョブ J_j が機械 M_i に処理されるときの機械の速度を s_{ij}
  無相関並列機械における処理時間 p_{ij} は p_j/s_{ij}

6　直列機械の分類

オペレーション O_{ij} の処理時間を p_{ij}，オペレーション O_{ij} を処理する機械を μ_{ij}

- **フローショップ（flow shop）：各ジョブのオペレーション数が機械の台数 m と一致し，各ジョブ J_j のオペレーションが $O_{1j}, O_{2j}, \cdots, O_{mj}$ の順で処理されなければならず，かつ各オペレーションが処理される機械がすべて異なり，かつその順序が同一のとき**
- **オープンショップ（open shop）：また，各ジョブ J_j のオペレーションが処理される機械がすべて異なり，かつ J_j のオペレーション $O_{1j}, O_{2j}, \cdots, O_{mj}$ の処理順があらかじめ決められていないとき**
- **ジョブショップ （job shop）：ジョブ J_j のオペレーション数を任意の正数とし m_j と記す．ジョブ J_j に対して，オペレーション $O_{1j}, O_{2j}, \cdots, O_{m_j j}$ の順序が与えられ，オペレーション O_{ij} を処理するための機械 μ_{ij} が任意のとき**

7　ジョブの属性

リリース時刻（release time, ready time, release date）r_j **：ジョブ** J_j **の処理を開始することができる最早時刻**

納期（due date）d_j **：ジョブ** J_j **の処理が終了することが望ましい時刻**

重み（weight）w_j **：ジョブ** J_j **の重要度**

分割可能（preemptive）**：ジョブが途中で中断でき，再び処理の途中から再開できるとき**
分割不可能（nonpreemptive）**：ジョブが分割可能でないとき**

8　先行制約

ジョブ間に先行順序が定義される場合がある．ジョブ J_j の処理完了後でないと，ジョブ J_k の処理が開始できないとき J_j, J_k 間に先行順序関係（precedence relation**）が存在するとよび，$J_j \to J_k$，もしくは $J_j \prec J_k$ と書く．**

9　目的関数

完了時刻（completion time）C_j ：ジョブ J_j が機械の時間軸を占有する最も遅い時刻
ジョブ J_j の納期 d_j
納期外れ（lateness）L_j：$L_j = C_j - d_j$
納期遅れ （tardiness）T_j ：$T_j = \max\{0, C_j - d_j\}$
納期遅れ単位ペナルティ（unit penalty）：

$$U_j = \begin{cases} 0 & C_j \leq d_j \text{ のとき} \\ 1 & \text{それ以外} \end{cases}$$

最大基準
メイクスパン（makespan）：$C_{\max} = \max_{j=1,\cdots,n} C_j$
最大納期外れ $L_{\max}$ ：$\max_{j=1,\cdots,n} L_j$
$L_{\max}$ を最小にするスケジュールは，最大納期遅れ $\max_{j=1,\cdots,n} T_j$ および $\max_{j=1,\cdots,n} U_j$ を最小にする
合計基準
総完了時刻，総滞留時間，フロー時間（flow time）$\sum C_j$ ： $\sum_{j=1}^{n} C_j$ （重みつき総完了時刻 $\sum w_j C_j$ ）
総納期遅れ$\sum T_j$：$\sum_{j=1}^{n} T_j$ （重みつき総納期遅れ $\sum w_j T_j$ ）
総納期遅れジョブ数$\sum U_j$ ：$\sum_{j=1}^{n} U_j$ （重みつき総納期遅れジョブ数 $\sum w_j U_j$ ）

10　意思決定レベルによる分類

生産計画（production planning）

- **システムデザインモデル**
  洞察を得るためのモデル
- **ストラテジックモデル**
  長期（1年から数年，もしくは数十年）の意思決定を支援するモデル
- **タクティカルモデル**
  中期（1週間から数ヶ月，もしくは数年）の意思決定を支援するモデル
- **オペレーショナルモデル**
  短期（リアルタイム，日ベース，もしくは週ベース）の意思決定を支援するモデル
  生産スケジューリング（production scheduling）
  ディスパッチング（dispatching）

11　構成要素による分類

$\alpha = (\alpha_1\alpha_2\alpha_3)$	$\beta = (\beta_1\beta_2\beta_3\cdots)$	γ
（資源，機械の特性）	（ジョブの特性）	（目的関数の情報）

α, β, γ の該当する欄が空白：$\circ$ の記号

将来事象に対する情報

- 事前に確定値がわかっている．（確定的，決定的: deterministic）
- 事前に確定値がわかっていないが，確率的な情報が与えられている．（確率的: stochastic, probabilistic）
- それ以外．（何の情報も与えられていない．）
  オンラインスケジューリング（on line scheduling）

12　資源（機械）の特性

資源（機械）の特性は $\alpha_1\alpha_2\alpha_3$
α_1 **は資源（機械）の機能**
α_2 **は資源（機械）の数**
α_3 **は資源（機械）の時間軸上での利用可能状況**
資源（機械）の機能

- **再生可能資源**（renewable resource）（$\alpha_1 =$ PS 1）
- **再生不可能資源**（nonrenewable resource）（$\alpha_1 =$ PS T）
  T **は計画時間の長さ**
- **再生可能資源と再生不可能資源の両者**（$\alpha_1 =$ PS $1T$）
- **部分再生（不）可能資源**（partially (non)renewable resource）（$\alpha_1 =$ PS var）

13　機械スケジューリングモデル

- **単一機械**（single machine）$(\alpha_1 = \circ)$
- **並列機械**（parallel machines）
  - **同一並列機械**（identical parallel machines）$(\alpha_1 = P)$
  - **一様並列機械**（uniform parallel machines）$(\alpha_1 = Q)$
  - **無相関並列機械**（unrelated parallel machines）$(\alpha_1 = R)$
- **直列機械**（serial machines）
  - **フローショップ**（flow shop）$(\alpha_1 = F)$
  - **オープンショップ**（open shop）$(\alpha_1 = O)$
  - **ジョブショップ**（job shop）$(\alpha_1 = J)$
  - **一般ショップ**（general shop）$(\alpha_1 = X)$
- **多機能機械**（multipurpose machine）
  - **同一**（identical）**並列機械** $(\alpha_1 = PMPM)$
  - **一様**（uniform）**並列機械** $(\alpha_1 = QMPM)$
  - **無相関**（unrelated）**並列機械** $(\alpha_1 = RMPM)$
  - **フローショップ**（flow shop）$(\alpha_1 = FMPM)$
  - **オープンショップ**（open shop）$(\alpha_1 = OMPM)$
  - **ジョブショップ**（job shop）$(\alpha_1 = JMPM)$
  - **一般ショップ**（general shop）$(\alpha_1 = XMPM)$

14　資源（機械）の数

- 問題として考えたとき，資源（機械）の数 m があらかじめ定数として与えられている．$(\alpha_2 = m)$
- 問題として考えたとき，資源（機械）の数は任意の正数であり，問題例が与えられたとき，はじめて定数として決められる．$(\alpha_2 = \circ)$

資源（機械）の利用可能状況

- 資源（機械）は計画期間の間は常に使用可能であり，資源の場合にはその使用可能量が一定である．($\alpha_3 = \circ$)
- 資源（機械）の利用可能状況が時刻によって異なる．($\alpha_3 =$ var)
  ここで var は “variable amount” を意味する．

15　ジョブの特性(1)

ジョブに対する属性は $\beta_1\beta_2\cdots$ と表現される．β_i の順序は任意であり，属性の数だけ付加される．

- ジョブの途中中断の可否
  - 分割可能（preemptive）$(\beta_1 = \text{pmtn})$
    ジョブまたはオペレーションが途中で中断でき，中断した箇所から再開することができる．ここで pmtn は “preemption” を意味する．
  - 分割不可能 (nonpreemptive）$(\beta_1 = \circ)$
    ジョブまたはオペレーションが途中で中断できない．
- ジョブ間の先行順序
  - 先行順序なし $(\beta_2 = \circ)$
  - 任意 $(\beta_2 = \text{cpm}$ もしくは prec$)$
  - 最小遅延時間（minimal lag time）つき先行順序 $(\beta_2 = \text{min})$
    - 開始-開始型の先行順序関係（start-start precedence relation）
    - 開始-終了型の先行順序関係（start-finish precedence relation）
    - 終了-終了型の先行順序関係（finish-finish precedence relation）
  - 最小・最大遅延時間（minimal and maximal lag time）つき先行順序 $(\beta_2 = \text{gpr})$
  - 有向根付木 $(\beta_2 = \text{tree})$
  - 鎖 $(\beta_2 = \text{chain})$

16　ジョブの特性(2)

リリース時刻（release time, ready time, release date）

- **リリース時刻（ジョブの処理が開始できる時刻）がジョブごとに異なる．**$(\beta_3 = r_j)$
- **すべてのジョブのリリース時刻が同じ．**$(\beta_3 = \circ)$ **（この場合には，一般性を失うことなくリリース時刻を** 0 **と仮定できる．）**

納期（due date, deadline）

- **納期（ジョブの処理が完了しなければならない時刻）がジョブごとに異なる．**$(\beta_5 = d_j)$
- **すべてのジョブの納期が同一である．**$(\beta_5 = d)$
- **納期が設定されていない．**$(\beta_5 = \circ)$

バッチ処理

- **直列バッチ**（serial batching）$(\beta_6 =$ s-batch$)$
  バッチ処理の終了時刻は，バッチに含まれるジョブの処理時間の和に等しく，バッチ処理を開始するには一定の段取り時間が必要であると仮定する．
- **並列バッチ**（parallel batchning）$(\beta_6 =$ p-batch$)$
  バッチ処理の終了時刻は，バッチに含まれるジョブの最も遅い完了時刻に等しい．

17　ジョブの特性(3)

多機械ジョブ（multiprocessor task）

- **線形加速** $(\beta_7 = \text{spdp-lin})$
- **機械台数の上限つき線形加速** $(\beta_7 = \text{spdp-lin-}\delta_j)$
- **任意加速** $(\beta_7 = \text{spdp-any})$
- **固定台数処理** $(\beta_7 = \text{size}_j)$
- **ハイパーキューブ型並列計算機** $(\beta_7 = \text{cube}_j)$
  ジョブ J_j **は，かならず** $\text{size}_j \in \{1, 2, 4, 8, \cdots\}$ **台の機械によって共同処理される．**
- **指定機械による処理 （固定型）** $(\beta_7 = \text{fix}_j)$
  ジョブ J_j **は，かならず機械の集合** fix_j **によって共同処理される．**
- **指定機械による処理 （選択型）** $(\beta_7 = \text{set}_j)$
- **多機械ジョブなし** $(\beta_7 = \circ)$

オペレーション間の待ちの可否

- **オペレーション間の待ち時間を許さない．** $(\beta_9 = \text{no-wait})$
- **オペレーション間の待ち時間を許す．** $(\beta_9 = \circ)$

18　ジョブの特性（資源制約つきスケジューリング）(1)

- 必要とする資源量
  - ジョブの遂行中は，一定量の資源を必要とする．$(\beta_{10} = \circ)$
  - ジョブの遂行中に必要とする資源量が変化してもよい．$(\beta_{10} = \mathsf{var})$
  - 資源の必要量が，ジョブの作業時間の関数となっている．
    - 離散関数（作業時間が離散量をとり，個々のとりえる値に対して資源の必要量が与えられている．）$(\beta_{10} = \mathsf{disc})$
    - 連続関数 $(\beta_{10} = \mathsf{cont})$
- モード
  - ジョブを遂行する手段（モード）が複数あり，ジョブはそれらから1つを選択することによって遂行される．$(\beta_{11} = \mathsf{mult})$
  - ジョブを遂行するモードが単一である．$(\beta_{11} = \circ)$
  - ジョブ集合が複数の部分集合に分割されており，各分割に含まれるジョブは同じモードで処理しなければならない．$(\beta_{11} = \mathsf{id})$；

19　ジョブの特性（資源制約つきスケジューリング）(2)

$\beta_{12} = \mathsf{res}\lambda\sigma\rho$

- 資源制約の（種類の）数
  - 問題として考えたとき，定数 λ として与えられている．($\lambda =$ 定数)
  - 問題として考えたとき，資源制約の数は任意の正数であり，問題例が与えられたとき，はじめて定数として決められる．($\lambda = \circ$)
- 各時刻における資源量の上限
  - 問題として考えたとき，各時刻における資源量の上限が定数 σ として与えられている．($\sigma =$ 定数)
  - 問題として考えたとき，資源量の上限は任意の正数であり，問題例が与えられたとき，はじめて定数として決められる．($\sigma = \circ$)
- ジョブが必要とする資源量
  - 問題として考えたとき，ジョブが必要とする資源量が定数 ρ として与えられている．($\rho =$ 定数)
  - 問題として考えたとき，ジョブが必要とする資源量は任意の正数であり，問題例が与えられたとき，はじめて定数として決められる．($\rho = \circ$)

20 目的関数 (1)

目的関数に対する属性は γ
正規基準（regular measure）　$\gamma = \text{reg}$ ：ジョブの完了時刻に対する非減少な目的関数
非正規基準（nonregular measure）$\gamma = \text{nonreg}$ ：ジョブの完了時刻に対する任意の目的関数

- **最大基準**
  - **最大完了時刻，メイクスパン** $(\gamma = C_{\max})$
  - **最大納期外れ** $(\gamma = L_{\max})$
  - **メイクスパン，最大納期外れを含む一般的な最大基準** $(\gamma = f_{\max})$
- **合計基準**
  - **総完了時刻** $(\gamma = \sum C_j)$
  - **総納期遅れ** $(\gamma = \sum T_j)$
  - **総納期遅れジョブ数** $(\gamma = \sum U_j)$
  - **重みつき総完了時刻** $(\gamma = \sum w_j C_j)$
  - **重みつき総納期遅れ** $(\gamma = \sum w_j T_j)$
  - **重みつき総納期遅れジョブ数** $(\gamma = \sum w_j U_j)$
  - **（重みつき）総完了時刻，（重みつき）総納期遅れ，（重みつき）総納期遅れジョブ数を含む一般的な合計基準** $(\gamma = \sum f_j)$

21 目的関数 (2)

- **平準化基準**
  - **資源使用量の平均からのずれの自乗和** $(\gamma = \sum \text{sq.dev})$
  - **資源使用量の平均からのずれの絶対値和** $(\gamma = \sum \text{abs.dev})$
  - **資源使用量の変化量の絶対値（ジャンプ）の重みつき和** $(\gamma = \sum \text{jump})$
- **トレードオフ基準**
  資源を使用したことによる費用と最大完了時刻のトレードオフを求める． $(\gamma = \sum \text{curve})$
- **財務的基準**
  - **ジョブを遂行することによるキャッシュフローの増加を現在価値（**net present value**）に変換したものの最大化** $(\gamma = \sum \text{npv})$
- **確率的基準**
  - **期待値基準期待値を表す** E **をつけて記す．**
  - **ジョブの作業時間や先行順序の確率的な情報下における最大完了時刻の累積密度関数** $(\gamma = \text{cdf})$
  - **ジョブが最長路（クリティカルパス）に含まれる確率（臨界指標：**criticality index**）を求める．** $(\gamma = \text{ci})$
  - **生起確率最大の最長路（クリティカルパス）を求める．** $(\gamma = \text{mci})$
- **その他の基準** $(\gamma = X)$
  たとえば上の基準の混合型（多目的計画）や機械の稼働率など

22　リリース時刻つき納期外れ最小化1機械スケジューリング問題

$1|r_j|L_{\max}$

単一の機械で n 個のジョブを処理する問題を考える．この機械は一度に1つのジョブしか処理できず，ジョブの処理を開始したら途中では中断できないものと仮定する．ジョブの集合を $\mathcal{J}$ ，その添え字を $1, 2, \cdots, n$ と書く．各ジョブ $j \in \mathcal{J}$ に対する作業時間 p_j，リリース時刻（ジョブの処理が開始できる最早時刻）r_j，および納期 d_j が与えられたとき，各ジョブの作業完了時刻 C_j の納期外れ L_j $(= C_j - d_j)$ の最大値を最小にするようなジョブを機械にかける順番（スケジュール）を求める．

$\mathcal{NP}$-困難

すべてのジョブに対してリリース時刻 r_j が同一のときには，納期の小さい順にジョブを機械にかける方法 EDD（earliest due date）が最適解

ジョブの作業を途中で中断することを許す問題は，分割可能リリース時刻つき納期外れ最小化1機械スケジューリング問題 $1|\text{pmtn}, r_j|L_{\max}$ とよばれ，やはり多項式時間で最適解

23　ジョブショップスケジューリング問題

$J||C_{\max}$

ジョブを $J_1, J_2, \cdots, J_n$，ジョブ J_j に属するオペレーションを $O_{1j}, O_{2j}, \cdots, O_{m_j j}$，機械を $M_1, M_2, \cdots, M_m$ とする．オペレーションは $O_{1j}, O_{2j}, \cdots, O_{m_j j}$ の順で処理されなければならず，オペレーション O_{ij} を処理するには機械 μ_{ij} を作業時間 p_{ij} だけ占有する．オペレーションが途中で中断できないという仮定の下で，最後のオペレーションが完了する時刻を最小化する各機械上でのオペレーションの処理順序を求める．

24　資源制約つきスケジューリング問題

PS $1|\text{cpm}|C_{\max}$
ジョブ（活動）の集合 $\mathcal{J}$ ，各時間ごとの使用可能量の上限をもつ資源 $\mathcal{R}$ が与えられている．ジョブとは，資源の一定量を使用することによって行う作業を指す．資源は，ジョブごとに決められた作業時間の間はジョブによって使用されるが，作業完了後は，再び使用可能になる．（このような資源を再生可能資源とよぶ．）ジョブ間に与えられた先行順序関係を満たした上で，最後のジョブの作業完了時刻を最小化するような，資源のジョブへの割り振りおよびジョブの作業開始時刻を求める．

資源制約つきスケジューリング問題 PS $1|\text{cpm}|C_{\max}$ **において，資源制約がないと仮定した問題** $|\text{cpm}|C_{\max}$ **（もしくは** $|\text{prec}|C_{\max}$**）は，**PERT/CPM**型のプロジェクト・スケジューリング問題とよばれる．PERT**（program evaluation and review technique）**およびCPM**（critical path method；**クリティカルパス法）は，スケジューリング理論の始祖とも言える古典的なモデル**

25　Gantt図式

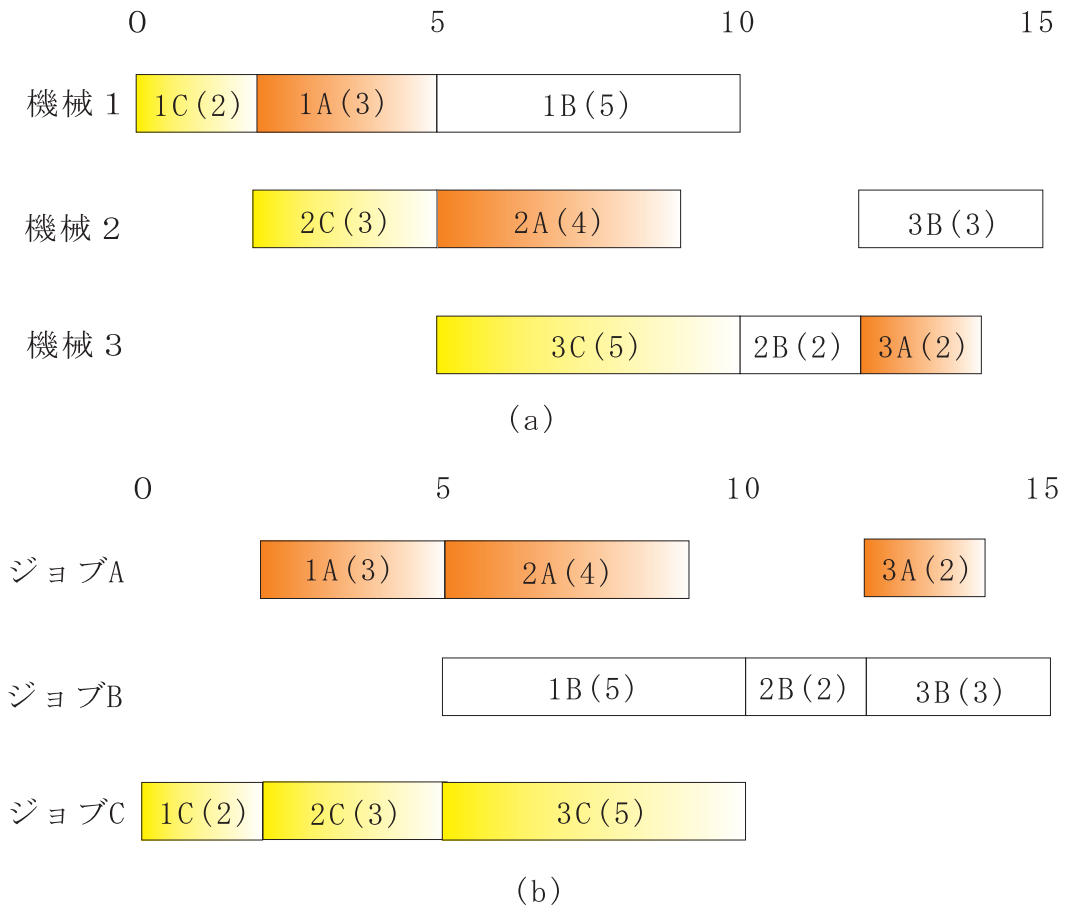


図1　3 ジョブ 3 機械のジョブショップスケジューリング問題に対するGantt図式.

26　点上活動図式と枝上活動図式

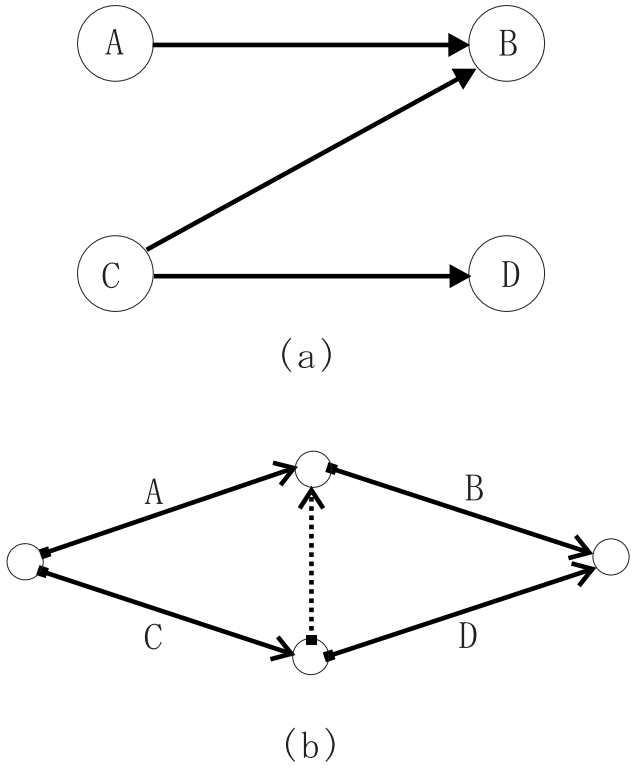


図2　点上活動図式と枝上活動図式.

27　離接グラフ表現

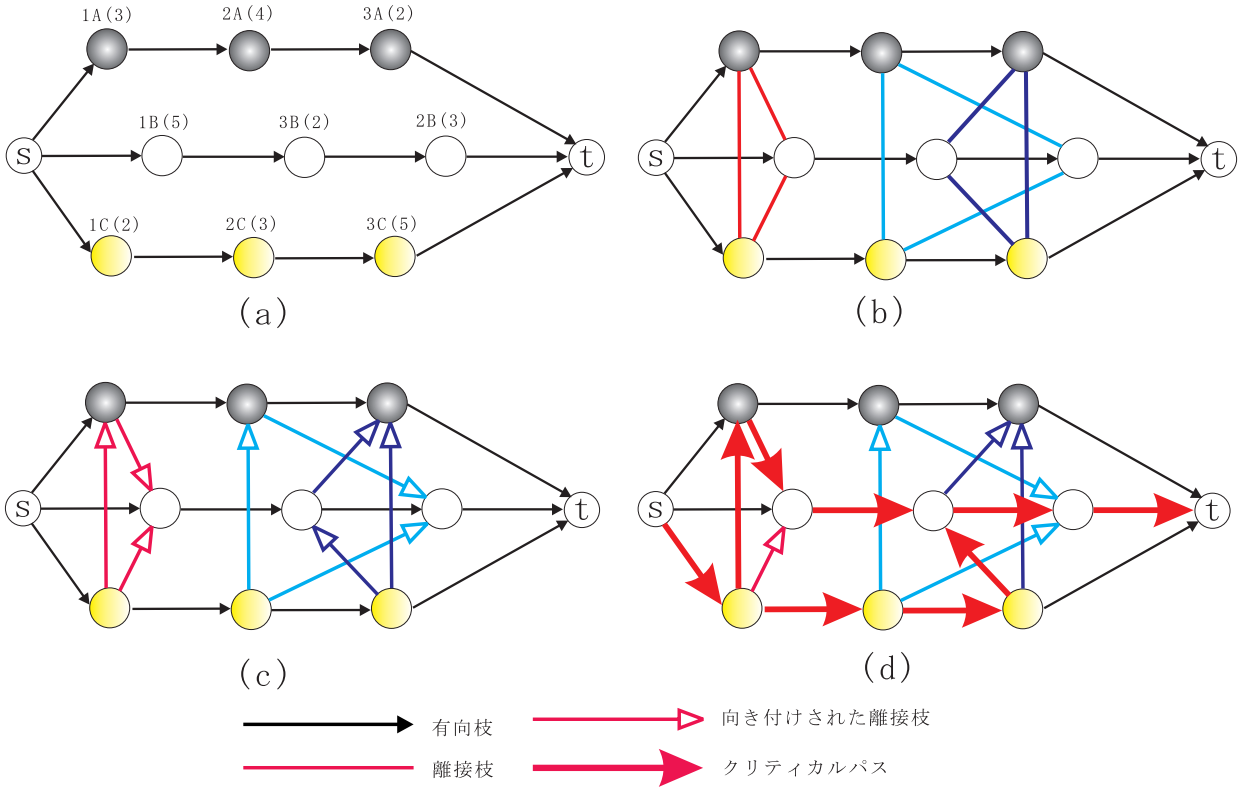


図3　3 ジョブ 3 機械のジョブショップスケジューリング問題に対する離接グラフ表現．この例では，2 本のクリティカルパスが存在する．

リリース時刻つき重みつき完了時刻和最小化 1 機械スケジューリング問題 $1|r_j|\sum w_jC_j$

単一の機械で n 個のジョブを処理する問題を考える．この機械は一度に1 つのジョブしか処理できず，ジョブの処理を開始したら途中では中断できないものと仮定する．ジョブの集合を $\mathcal{J}$ ，その添え字を $1,2,\cdots,n$ と書く．各ジョブ $j\in\mathcal{J}$ に対して，リリース時刻 $r_j\in\mathsf{Z}_+\cup\{0\}$，作業時間 $p_j\in\mathsf{Z}_+$ と重要度を表す重み $w_j\in\mathsf{R}_+$ が与えられたとき，各ジョブ j の作業完了時刻 C_j の重みつきの和を最小にするようなジョブを機械にかける順番（スケジュール）を求める．

時間の離散化：適当な時刻 $0,1,\cdots,T$ で時間を区切り，ある時刻からある時刻までの間を期（period）とよぶことにする．

連続時間との対応づけは，時刻 $t-1$ から時刻 t までを期 t とよぶものとする．離散化した時刻 $0,1,\cdots,T$ に対して，期の集合を $\mathcal{T}=\{1,2,\cdots,T\}$ と定義する．

29 x-定式化

ジョブ j が期 t に開始するとき 1，それ以外のとき 0 になる 0-1 変数 x_{jt} を用いる．

$$\text{minimize} \quad \sum_{j \in \mathcal{J}} \sum_{t=r_j+1}^{T-p_j+1} w_j(t+p_j-1)x_{jt}$$

$$\text{subject to} \quad \sum_{t=r_j+1}^{T-p_j+1} x_{jt} = 1 \qquad \forall j \in \mathcal{J}$$

$$\sum_{j \in \mathcal{J}} \sum_{s=t-p_j+1}^{t} x_{js} \leq 1 \qquad \forall t \in \mathcal{T}$$

$$x_{jt} \in \{0,1\} \qquad \forall j \in \mathcal{J}, t = r_j+1, \cdots, T-p_j+1$$

30 y-定式化

ジョブ j が期 t に処理されているとき 1，それ以外のとき 0 を表す 0-1 変数 y_{jt} を用いる．

$$\text{minimize} \quad \sum_{j\in\mathcal{J}} w_j \left\{ \frac{p_j}{2} + \frac{1}{p_j} \sum_{t=r_j+1}^{T} \left(t-\frac{1}{2}\right) y_{jt} \right\}$$

$$\text{subject to} \quad \sum_{t=r_j+1}^{T} y_{jt} = p_j \qquad \forall j \in \mathcal{J}$$

$$\sum_{j\in\mathcal{J}} y_{jt} \le 1 \qquad \forall t \in \mathcal{T}$$

$$y_{jt} \in \{0,1\} \qquad \forall j \in \mathcal{J}, t = r_j+1, \cdots, T$$

目的関数内の $t-\frac{1}{2}$ は，時刻 t の中間の時間なので，$(1/p_j)\sum_{t=r_j+1}^{T}\left(t-\frac{1}{2}\right)y_{jt}$ の項は，ジョブ j が処理される時刻の重心を表す．それに処理時間の半分 $p_j/2$ を加えることによって，完了時刻に相当する量を計算していることになる．

0-1 変数を 0 以上 1 以下に線形計画緩和した問題の下界は，x-定式化より弱い

31　$O(n \log n)$ 時間のアルゴリズム

y-定式化の線形計画緩和のための厳密解法

- **リリース時刻が 0 のジョブを優先キューに（w_j/p_j の非減少順になるように）入れる．**
- **時刻 0 から始めて，すべてのジョブがスケジュールされるまで，以下を繰り返す．**
  - **優先キューから w_j/p_j が最小のジョブを取り出し，そのジョブが完了するか，次にリリースされるジョブが発生するまで，機械で処理する．**
  - **このとき，ジョブが完了したなら，優先キューから除き，次にリリースされるジョブが発生したなら，w_j/p_j をキーとして優先キューに入れる．**

表1　$1|r_j|\sum w_jC_j$ の例題．

ジョブ番号 j	r_j	p_j	w_j
1	0	10	1
2	4	5	1
3	6	2	1
4	7	4	2

32　例題の解

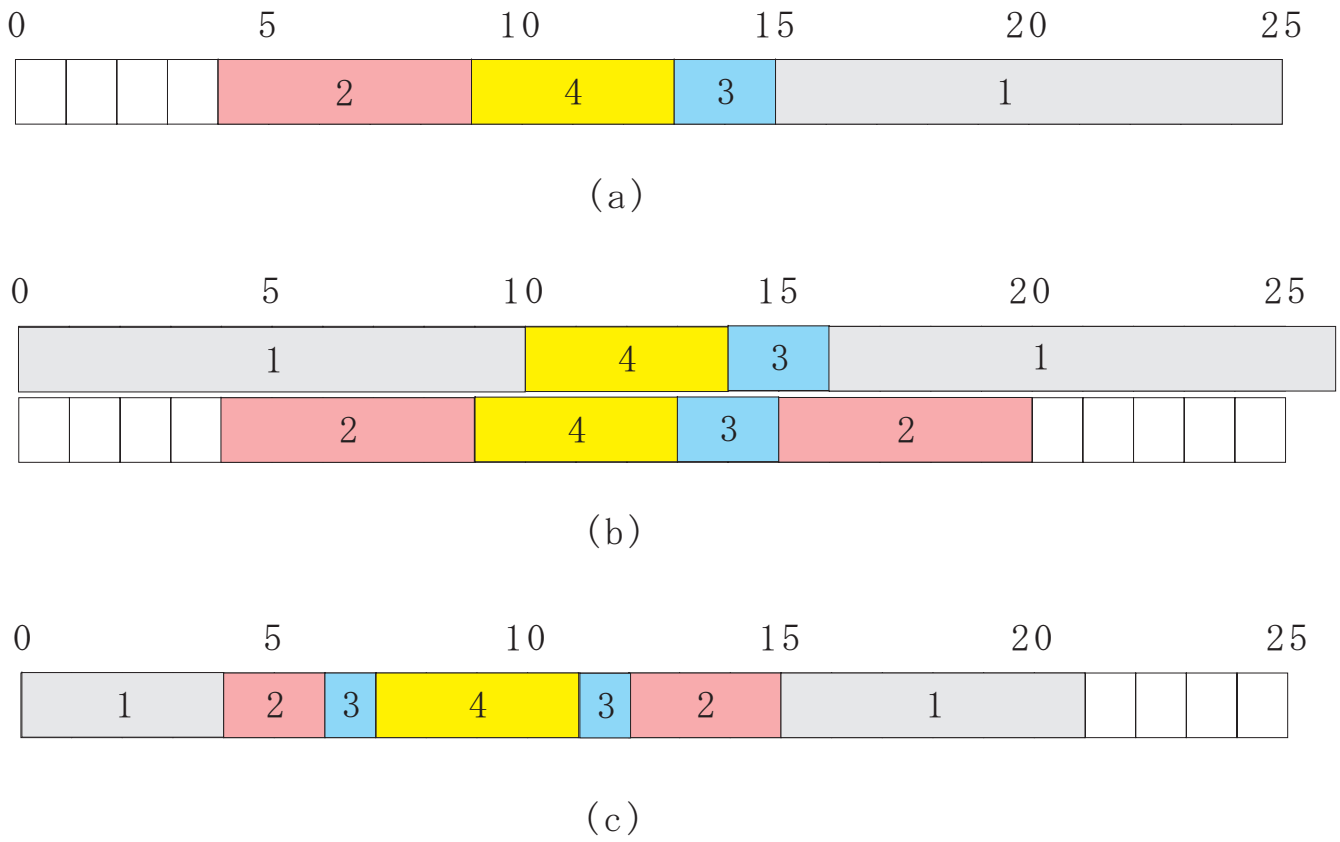


図4　$1|r_j|\sum w_jC_j$ **の例題の解．**(a) **最適解** (75)**．**(b) x-**定式化の緩和解** (75)**．**(c) y-**定式化の緩和解** (61.2)**．**

33　資源制約つきスケジューリング問題の定式化 (1)

基本モデルの定式化で用いる集合，入力データ，変数

集合

$\mathcal{J}$:　ジョブの集合．
$\mathcal{T}$:　期の集合．$\mathcal{T} = \{1, 2, \cdots, T\}$．連続時間との対応づけは，時刻 $t-1$ から t までを期 t と定義する．
$\mathcal{R}$:　資源の集合．
$\mathcal{P}$:　ジョブ間の先行順序関係を表す集合．$\mathcal{J} \times \mathcal{J}$ の部分集合で，$(j,k) \in \mathcal{P}$ のとき，ジョブ j の処理が終了するまで，ジョブ k の処理が開始できないことを表す．

入力データ

p_j:　ジョブ j の処理時間．非負の整数を仮定する．
c_{jt}:　ジョブ j を時刻 t に開始したときの費用．
R_{jr}:　ジョブ j の処理に要する資源 r の量．
RUB_{rt}:　時刻 t における資源 r の使用可能量の上限．

変数

x_{jt}:　ジョブ j を時刻 t に開始するとき 1，それ以外のとき 0 を表す 0-1 変数．

34　資源制約つきスケジューリング問題の定式化 (2)

$$\text{minimize} \sum_{j \in \mathcal{J}} \sum_{t=1}^{T-p_j+1} c_{jt} x_{jt}$$

subject to **ジョブ遂行条件**

資源制約

先行順序制約

変数の 0-1 **条件**

ジョブ遂行条件:

$$\sum_{t=1}^{T-p_j+1} x_{jt} = 1 \quad \forall j \in \mathcal{J}$$

資源制約:

$$\sum_{j \in \mathcal{J}} R_{jr} \sum_{s=\max\{t-p_j+1,1\}}^{\min\{t,T-p_j+1\}} x_{js} \leq RUB_{rt} \quad \forall r \in \mathcal{R}, t \in \mathcal{T}$$

35　資源制約つきスケジューリング問題の定式化 (3)

先行順序制約 1:

$$\sum_{s=1}^{t} x_{js} \geq \sum_{s=1+p_j}^{t+p_j} x_{ks} \quad \forall (j,k) \in \mathcal{P}, t = 1+p_j, \cdots, T-p_j \tag{1}$$

ジョブ j の処理が終了するまで，ジョブ k の処理が開始できないことを表す．

先行順序制約 2:

$$\sum_{t=2}^{T-p_j+1} (t-1)x_{jt} + p_j \leq \sum_{t=2}^{T-p_k+1} (t-1)x_{kt} \quad \forall (j,k) \in \mathcal{P}$$

ジョブ j の処理が終了するまで，ジョブ k の処理が開始できないことを表す．左辺の式は，ジョブ j の完了時刻を表し，右辺の式はジョブ k の開始時刻の直前の期を表す．

変数の 0-1 **条件**

$$x_{jt} \in \{0,1\} \quad \forall j \in \mathcal{J}, t = 1, \cdots, T-p_j+1$$

36　スケジュール生成スキーム

正規完了時刻基準（ジョブの完了時刻に対する非減少な関数を目的関数）

定義 1 (有効スケジュール（active schedule）) 他のジョブの開始時刻を遅らせることなしに，いかなるジョブの開始時刻も早めることができないスケジュールを有効スケジュールとよぶ.

定義 2 (遅れなしスケジュール（nondelay schedule）) ジョブの作業の途中中断を許したとしても，他のジョブの完了時刻を遅らせることなしに，いかなるジョブの開始時刻を早めることができないスケジュールを遅れなしスケジュールとよぶ.

命題 1 正規完了時刻基準をもつスケジューリング問題を考える．このとき，有効スケジュールの中に必ず最適スケジュール（の内の 1 つ）が存在するが，遅れなしスケジュールは最適スケジュールを含むとは限らない.

37　有効スケジュール生成スキーム

すでにスケジュールされたジョブをスケジュール済みジョブ（scheduled job）とよび，その集合を $\mathcal{S}$
$\mathcal{J} \setminus \mathcal{S}$ に含まれており，かつすべての先行するジョブがすでにスケジュールされたジョブを適合ジョブ（eligible job）とよび，その集合を $\mathcal{E}$

- **スケジュール済みジョブの集合 $\mathcal{S}$ を $\emptyset$，資源の使用量をすべての時刻に対して 0 と初期設定する．**
- **すべてのジョブがスケジュールされるまで，以下を繰り返す．**
  - **適合ジョブ集合 $\mathcal{E}$ から 1 つのジョブ j を選択する．**
  - **先行順序制約と資源制約を満たす最早時刻をジョブ j の作業開始時刻とする．**
  - **$\mathcal{S}$ に j を追加し，各時刻における資源の使用量を更新する．**

38　遅れなしスケジュール生成スキーム

完了ジョブ（complete job）$\mathcal{C}(t)$ ：時刻 t までに作業が完了しているジョブ
活動中ジョブ（active job）$\mathcal{A}(t)$：時刻 t において作業中のジョブ
時刻 t において，すべての先行ジョブが完了ジョブになっており，かつ資源制約を破らないようにスケジュール可能なジョブで $\mathcal{J} \setminus (\mathcal{C} \cup \mathcal{A})$ に属するものを時刻 t における適合ジョブとよび，その集合を $\mathcal{E}(t)$

- **完了ジョブの集合 $\mathcal{C}$ および活動中ジョブの集合 $\mathcal{A}$ を $\emptyset$，現在時刻 t を 0，資源の使用量をすべての時刻に対して 0 と初期設定する．**
- **すべてのジョブの作業開始時刻が決定されるまで，以下を繰り返す．**
  - **もし，$\mathcal{E}(t)$ が空ならば，$\mathcal{E}(t)$ に入るジョブができるまで，t を増やす．**
  - **時刻 t における適合ジョブ集合 $\mathcal{E}(t)$ から 1 つのジョブ j を選択する．**
  - **ジョブ j の作業開始時刻を t に設定する．**
  - **活動ジョブの集合 $\mathcal{A}$ に j を追加し，各時刻における資源の使用量を更新する．**
  - **$\mathcal{A}$ 内のジョブで作業終了時刻が最早のものを j' とする．**
  - **j' を $\mathcal{A}$ から除き，$\mathcal{C}$ に加える．**
  - **現在時刻 t をジョブ j' の作業完了時刻に更新する．**

39　有効（遅れなし）スケジュール生成スキームの例

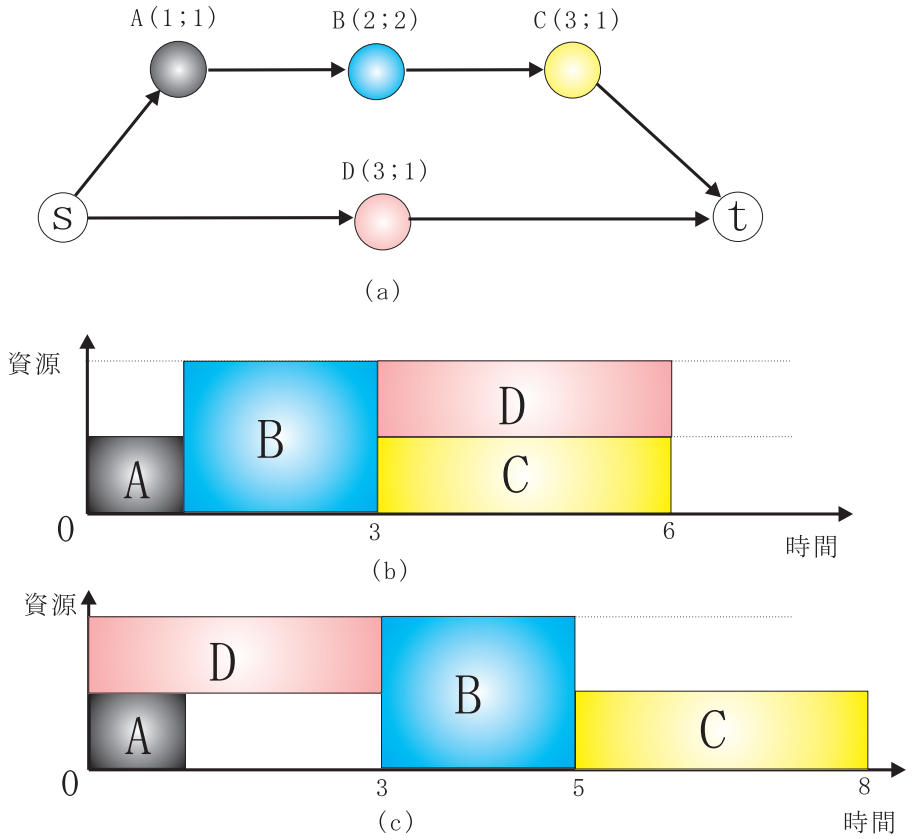


図5　(b) 有効スケジュール生成スキームによる解．(c) 遅れなしスケジュール生成スキームによる解．

40　優先ルール

優先ルール（priority rule, dispatching rule**；ディスパッチングルール）は，適合ジョブ集合** $\mathcal{E}$ **（もしくは** $\mathcal{E}(t)$**）から** 1 **つのジョブ** j **を選択するための “ルール”**

SPTルール（shortest processing time rule**）：処理時間** p_j **が最小になるジョブ** j **を選択**

WSPTルール（weighted shortest processing time rule**）：ジョブの重要度** w_j **と作業時間** p_j **の商** w_j/p_j **を最大化するジョブを選択**

EDDルール（earliest due date rule**）もしくはJacksonルール（**Jackson's rule**）：納期** d_j **の早い順**

その他にも様々な（複合）ルール

41　近傍

近傍（neighborhood）：解の集合から解のべき集合への写像
局所探索法をはじめとする多くのメタ解法の基礎
ジョブショップスケジューリング問題 $J\,||\,C_{\max}$ に対する近傍

定義 3 (離接枝反転近傍) 離接枝反転近傍とは，与えられた実行可能解に対応するクリティカルパス $P(s,t)$ 上の離接枝を反転することによって得られる解の集合を指す.

離接枝反転近傍において離接枝をクリティカルパス上に限定したことには，以下の2 つの利点がある.

- **クリティカルパス上にない離接枝は，反転したときに最大完了時刻（メイクスパン）を減少させることはない，**
- **クリティカルパス上の離接枝を反転して得られる離接グラフは閉路をもたない.**

42　制限つき離接枝反転近傍

ジョブ上で u の直前に処理される点：ジョブ先行点（job-predecessor**）** $\alpha(u)$
直後に処理される点：ジョブ後続点（job-successor**）** $\gamma(u)$
機械上における u の直前の点：機械先行点（machine-predecessor**）** $\beta(u)$
直後に処理される点：機械後続点（machine-successor**）とよび** $\delta(u)$

定義 4 (制限つき離接枝反転近傍) 制限つき離接枝反転近傍とは，与えられた実行可能解に対応するクリティカルパス $P(s,t)$ 上の離接枝 (u,v) で，u のジョブ先行点 $\alpha(u)$ もしくは v のジョブ後続点 $\gamma(v)$ のいずれか一方が $P(s,t)$ に含まれているものを反転することによって得られる解の集合を指す.

命題 2 与えられた実行可能解に対応するクリティカルパス $P(s,t)$ 上の離接枝 (u,v) で，u のジョブ先行点 $\alpha(u)$ と v のジョブ後続点 $\gamma(v)$ のいずれも $P(s,t)$ に含まれていないものとする．このとき，離接枝 (u,v) を反転することによって最大完了時刻（メイクスパン）は改善されない.

43　並列機械スケジューリング問題

MIP/CP アプローチ（MIP/CP approach）$=$ **混合整数計画**（Mixed Integer Programming: MIP）$+$ **制約論理**（Constraint Programming: CP）

リリース時刻と納期つきの並列機械スケジューリング問題 $R|r_j d_j|\sum f_j$
機械の総数を m **と書き，機械の添え字集合を** $\{1,2,\cdots,i,\cdots,m\}=\mathcal{M}$
ジョブの総数を n **と書き，ジョブの添え字集合を** $\{1,2,\cdots,j,\cdots,n\}=\mathcal{J}$
ジョブ j **が作業を開始できる最早の時刻をリリース時刻もしくは最早開始時刻**（earliest start time）**とよび，** EST_j **と記す．**
ジョブ j **が作業が終了しなければならない最遅の時刻を納期もしくは最遅終了時刻**（latest completion time）**とよび，**LCT_j **と記す．**
ジョブ j **を機械** m **で処理するときの費用を** c_{jm}**，処理時間を** p_{jm}

44 $R|r_j d_j|\sum f_j$ の定式化

ジョブ j を機械 m で処理するとき 1，それ以外のとき 0 を表す 0-1 変数 x_{jm}

$$\text{minimize} \quad \sum_{j\in\mathcal{J}}\sum_{m\in\mathcal{M}} c_{jm}x_{jm}$$

$$\text{subject to} \quad \sum_{m\in\mathcal{M}} x_{jm} = 1 \qquad \forall j \in \mathcal{J}$$

機械 m 上に割り振られたジョブが処理可能 $\forall m \in \mathcal{M}$ (2)

$$x_{jm} \in \{0,1\} \qquad \forall j \in \mathcal{J}, m \in \mathcal{M}$$

式(2)の判定に制約論理を用いて，もし実行不能ならば，以下の実行不能カットを付け加える．

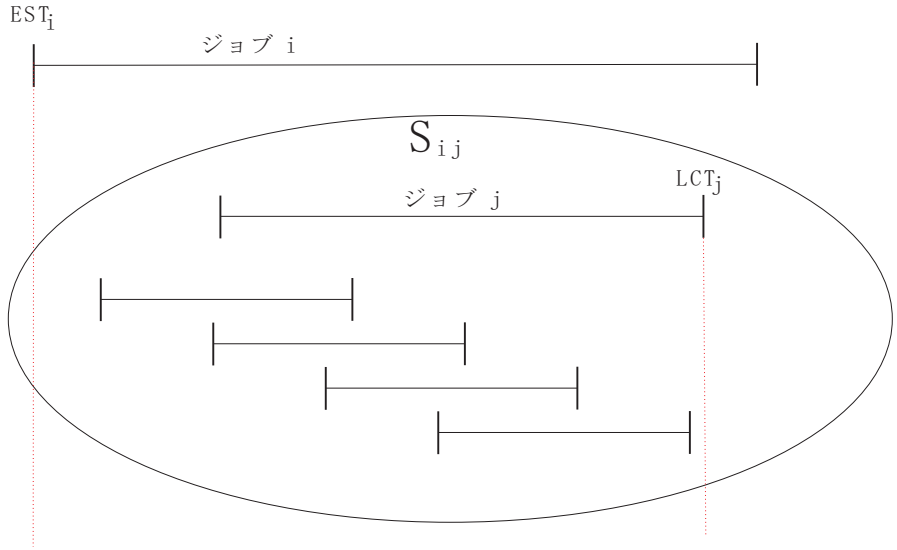
$$\sum_{j\in\mathcal{J}_m} x_{jm} \leq |\mathcal{J}_m| - 1$$

ここで，$\mathcal{J}_m$ は，線形計画緩和の解 $\bar{x}$ を与えたとき，ある機械 m に対して $\sum_{j\in\mathcal{J}_m} \bar{x}_{jm} > |\mathcal{J}_m| - 1$ となっているジョブの部分集合

45　実行不能カットの強化

リリース時刻が EST_i **以降で，納期が** LCT_j **以下のジョブの集合を** S_{ij}

$$S_{ij} = \{k \in \mathcal{J} \mid \mathsf{EST}_i \leq \mathsf{EST}_j, \mathsf{LCT}_k \leq \mathsf{LCT}_j\}$$



妥当不等式

$$\sum_{k \in S_{ij}} p_{km} x_{km} \leq \mathsf{LCT}_j - \mathsf{EST}_i$$

$\Rightarrow$ 0-1 **ナップサック問題の制約**

46　持ち上げによる強化

左辺において，S_{ij} に含まれていないジョブ k に対して
納期 LCT_j からはみ出た時間 $(\mathsf{LCT}_k - \mathsf{LCT}_j)^+$ を β_{jk}
リリース時刻 EST_i の前にはみ出た時間 $(\mathsf{EST}_i - \mathsf{EST}_k)^+$ を α_{ik}
ジョブ k が機械 m に割り当てられたとき，$[\mathsf{EST}_i, \mathsf{LCT}_j]$ の間で使われる時間

$$\gamma_{km} = \min\{\mathsf{LCT}_j - \mathsf{EST}_i, p_{km} - \max\{\alpha_{ik}, \beta_{jk}\}\}$$

妥当不等式

$$\sum_{k \in \mathcal{J}} \gamma_{km} x_{km} \leq \mathsf{LCT}_j - \mathsf{EST}_i$$