青 山 学 院 大 学 理 工 学 部
経 営 シ ス テ ム 工 学 科

卒 業 論 文

並列機械モデルにおける
最大実行開始待ち時間最小化問題の計算論的分析

２０１７年度

天本　祐希
15713004

指導教員　宋 少秋　印

目 次

第1章　はじめに

1.1　研究背景

一般に，Web サービスを運用している会社では，運用しているサービスの応答が遅いと，顧客離れやクレームの被害を受けることがある．そのため，計算サーバーの応答の早さは重要である．応答の早い計算サーバーを作るためには，与えられたタスクをどのように割り当て，処理するかを考える必要がある．計算サーバーのタスクの処理開始は，サービスの利用者が，ネットワークを介して，そのサービスを運用している会社の計算サーバーにタスク処理の要求を行い，そのタスクが計算サーバーに到着した後である．

タスクの情報が Web サービス利用者に依存するため，タスク処理を行う計算サーバーは，割り当てるタスクの情報があらかじめわからない．このような環境をオンライン環境といい，オンライン環境において，スケジュールを決定する問題をオンラインスケジューリング問題という．対して，割り当てるタスクの情報すべてがあらかじめわかっている環境をオフライン環境という．

タスクの到着時刻とは，スケジューリング問題における処理開始可能時刻である．また，実行開始待ち時間とは，タスクが到着してから，タスクの処理が開始されるまでの時間である．最大実行開始待ち時間最小化問題は，処理開始可能時刻を制約とし，最大の実行開始待ち時間の最小化を目的とするスケジューリング問題である．

したがって，計算サーバーへのタスク割り当ては，最大の実行開始待ち時間の最小化を目的とするオンラインスケジューリング問題として捉えることができる．オフライン環境，オンライン環境における最大実行開始待ち時間を目的関数としたスケジューリング問題は未解決問題である．

1.2　研究目的

最大実行開始待ち時間最小化問題はどの機械モデルにおいても計算複雑さが明らかでない．本研究では，オフライン環境における最大実行開始待ち時間最小化問題に対して，以下を目的とする．

目的 1： 最大実行開始待ち時間最小化問題の計算複雑さを明らかにする．

最大実行開始待ち時間最小化問題はどの機械モデルにおいても，問題の計算複雑さは明らかでない．本研究では，機械モデル，機械数に着目し問題の難しさに影響を与える特徴を明らかにする．

目的 2： 最大実行開始待ち時間最小化問題に対する効率的解法の提案を行う．

各機械モデルにおける最大実行開始待ち時間最小化問題の計算複雑さに基づいて，解法の提案を行う．また，解法の適用先である計算サーバーにおいて，解法の実験的評価を行う．

1.3 研究成果

本研究では，最大実行開始待ち時間最小化問題に関して以下の成果を得た．

成果 1： 最大実行開始待ち時間最小化問題を決定問題として捉えたとき，無関連並列機械モデルにおいて機械数が入力の一部の場合，NP 完全であることを明らかにした．

最大実行開始待ち時間最小化問題 は JIT ジョブ荷重和最大化問題の拡張問題，処理開始可能時刻付き最大遅れ時間最小化問題の部分問題として捉えることができる．JIT ジョブ荷重和最大化問題は無関連並列機械モデルにおいて機械数が入力の一部の場合， 処理開始可能時刻付き最大遅れ時間最小化問題は単一機械モデルにおいて強 NP 困難であることが証明されている．最大実行開始待ち時間最小化問題とこれらの問題の対応を取り，3-SAT からの還元により最大実行開始待ち時間最小化問題の NP 完全性を証明した．

成果 2： 同一並列機械モデル，無関連並列機械モデルにおける最大実行開始待ち時間最小化問題に対するヒューリスティックを開発し，対象とする環境における提案解法の有効性を示した．

貪欲アルゴリズムに基づいたヒューリスティックを開発した．ヒューリスティックにより出力された解から得られた目的関数の値を W_h，分枝限定法により出力された最適解より得られた目的関数の値を W_{opt} とすると，異なるインスタンスに対して，W_h/W_{opt} を求めることで，提案解法の評価を行った．W_h/W_{opt} の値が 1 に近いほど，解法の質は良いと言える．本研究の実験によると，$\max\left\{W_h/W_{opt}\right\} =?$ の結果を得た．

成果 3： 同一並列機械モデル，無関連並列機械モデルにおける最大実行開始待ち時間最小化問題に対する厳密解法を開発し，解法に対する計算時間の評価を行い，実用性を示した．

NP 困難な最適化問題は多項式時間で最適解を求めることができないと考えられている．そのため，最適解を求めるためには，実行可能解を全列挙し比較する必要がある．そこで，本研究では，厳密解法に分枝限定法アルゴリズムを用いた．ジョブの機械への割り当てに対応する分割生成アルゴリズムに対して，分割の要素数 = 機械数 となる分割のみを生成するアルゴリズムに改良した．この改良により，考慮する分割の数を減らすことで計算効率を向上させた．また，処理開始可能時刻付き最大遅れ時間最小化問題の部分問題に対する多項式アルゴリズムの概念を導入した．この改良により，列挙する順列を減らし，計算効率を向上させた．その結果，同じインスタンスに対して，計算時間を約 ??? 倍にすることに成功した．

1.4 章構成

第 2 章では，最大実行開始待ち時間最小化問題の定式化を行なう．また，既存のスケジューリング問題との対応づけ，NP 完全性の証明のために，最大実行開始待ち時間最小化問題を決定問題として定義し，定式化を行う．

第 3 章では，最大実行開始待ち時間最小化問題に関連する，JIT ジョブ荷重和最大化問題と処理開始可能時刻付き最大遅れ時間最小化問題に対する従来研究の成果を，計算複雑さの観点，解法の観点から，それぞれまとめる．

第 4 章から第 5 章にかけて，本研究の成果を述べる．第 4 章では，最大実行開始待ち時間最小化問題の計算複雑さについて述べる．

第 5 章では，提案解法の紹介と，分析規模拡大のために改良した分枝限定法の紹介を行う．また，実験より得たデータから提案解法の評価を行う．

第 6 章では，結論として，本研究の成果と今後の課題を述べる．

第 7 章では，第 5 章で提案した解法の分析に用いた実験結果を載せる．

第2章　問題の定式化

ここでは，最大実行開始待ち時間最小化問題を定式化する．以下 2 つの定式化に共通する，機械モデルについて示す．

各ジョブを処理する機械の性能によって，機械モデルは次のように分類される．まず，ジョブを処理する機械の台数について，一つの機械で処理する単一機械モデルと，複数の機械で処理を行なう並列機械モデルに分類される．並列機械モデルはさらに，全ての機械の性能が等しい同一並列機械モデル，機械ごとに処理速度が異なる一様並列機械モデル，ジョブと機械の組み合わせによって処理時間が異なる無関連並列機械モデルに分類される．ここで，各機械モデル間の関係を示す．

単一機械モデルは，同一並列機械モデルにおいて，機械数 1 の場合と対応する．したがって，単一機械モデルは，同一並列機械モデルの一部として捉えることができる．

同様に，同一並列機械モデルは，一様並列機械モデルの一部として，一様並列機械モデルは無関連並列機械モデルの一部として捉えることができる．

2.1　最大実行開始待ち時間最小化問題

最大実行開始待ち時間最小化問題とは，各ジョブの処理開始可能時刻を制約とし，処理開始可能時刻と処理開始時刻の差の最大値の最小化を目的とするスケジューリング問題である．

以下では，無関連並列機械モデルについて，最大実行開始待ち時間最小化問題を定式化する．

入力： ジョブの集合を $\mathcal{J}$，無関連機械の集合 $\mathcal{M}$ と表記する，処理開始可能時刻関数 $r : \mathcal{J} \to \mathbb{N}$，処理時間関数 $p : \mathcal{J} \times \mathcal{M} \to \mathbb{N}$ であり，それぞれの関数が返す値は自然数である．入力は 4 項組，$(\mathcal{J}, \mathcal{M}, r, p)$．

解： 問題の前提に基づき，スケジュールを定式化する．スケジュールは以下の条件を満たす $A : \mathcal{J} \to \mathcal{M}$ と $s : \mathcal{J} \to \mathbb{N}$ の対 (A, s) であり，スケジュールによって，各ジョブを，どの機械で，いつ処理するかを決める．

- $\forall J \in \mathcal{J}\big[s(J) \geq r(J)\big]$
  - 各ジョブは処理開始可能時刻以降に処理を開始する
- $\forall J, J' \in \mathcal{J}\ \Big[\big[J \neq J' \wedge A(J) = A(J')\big] \Rightarrow [s(J), s(J)+p(J, A(J))) \cap [s(J'), s(J')+p(J', A(J'))) = \emptyset\Big]$
  - 各機械は同時に複数のジョブを処理しない．
  - 各ジョブの処理を開始すると，完了するまで中断しない．

目的関数： 実行可能なスケジュール (A, s) のうち，最大の実行開始待ち時間，

$$\varphi(A, s) = \max_{J \in \mathcal{J}} \{s(J) - r(J)\}$$

を最小とするスケジュールを求める．

2.2 最大実行開始待ち時間最小化問題（決定問題）

最大実行開始待ち時間最小化問題（決定問題）とは，各ジョブの処理開始可能時刻を制約とし，処理開始可能時刻と処理開始時刻の差の最大値が w 以下となるスケジュールが存在するかを判定する問題である．

以下では，無関連並列機械モデルについて，最大実行開始待ち時間最小化問題を決定問題として定義し定式化する．

インスタンス： ジョブの集合 $\mathcal{J}$ を無関連機械の集合 $\mathcal{M}$ と表記する．処理開始可能時刻関数 $r : \mathcal{J} \to \mathbb{N}$，処理時間関数 $p : \mathcal{J} \times \mathcal{M} \to \mathbb{N}$，実行開始待ち時間 w であり，それぞれの関数が返す値は自然数である．入力は 5 項組，$(\mathcal{J}, \mathcal{M}, r, p, w)$．

問題： 以下の条件を満たす $A : \mathcal{J} \to \mathcal{M}$ と $s : \mathcal{J} \to \mathbb{N}$ の対 (A, s) が存在するかどうか．

- $\forall J \in \mathcal{J}\big[s(J) \geq r(J)\big]$
  - 各ジョブは処理開始可能時刻以降に処理を開始する．
- $\forall J, J' \in \mathcal{J}\ \Big[\big[J \neq J' \wedge A(J) = A(J')\big] \Rightarrow [s(J), s(J)+p(J, A(J))) \cap [s(J'), s(J')+p(J', A(J'))) = \emptyset\Big]$
  - 各機械は同時に複数のジョブを処理しない．
  - 各ジョブの処理を開始すると，完了するまで中断しない．
- $\max\big\{s(J) - r(J) \mid J \in \mathcal{J}\big\} \leq w$
  - ジョブの処理開始可能時刻からその処理を開始するまでの待ち時間は w 以下．

2.2.1 複雑性クラス

決定問題は，インスタンスと問題によって定義され，決定問題は判定として Yes または no のいずれかを持つ．決定問題が定義されると，アルゴリズムによってどれくらい早くその問題を正しく解くことができるかを考える．一般に，アルゴリズムの計算時間は問題のインスタンスの大きさに比例する．あるアルゴリズムの計算時間がインスタンスの大きさの多項式関数ならば，そのアルゴリズムは効率的と呼ばれる．少なくとも一つ効率的なアルゴリズムが存在する問題のクラスは P で表される．

NP は問題のクラスの一つであり，NP は，問題のインスタンスについての判定が Yes ならば，その時，多項式サイズの証拠が存在し，その証拠が Yes の証拠であることが多項式時間で判定することができるすべての決定問題のクラスである．クラス NP は P をふくむ，そして一般的に NP と P はそれぞれ異なるクラスの問題として受け入れられている．NP に属する任意の問題と少なくとも同じくらい難しい問題を NP 困難 であるといい、そのうち NP に属するものを NP 完全問題という．

次に，決定問題の 1 つである 3-SATISFIABILITY (3-SAT) と 3-PARTITION を紹介する．

3-SATISFIABILITY (3-SAT) の定義は以下の通り．

インスタンス： ブール型の変数集合 $X = \{x_1, \ldots, x_n\}$ と，X 上の 3 つのリテラルからなる集合 h の集合 H．ただし，$H \subseteq 2^{L_X}$ s.t. $\forall h \in H\big[|h| = 3\big]$ を満たす．ここで，リテラルの集合とは $L_X = X \cup \{\bar{x} \mid x \in X\}$ で表すことができる集合である．

問題： H を充足する真理値割り当て $f : X \to \{0, 1\}$，つまり，

$$\bigwedge_{h \in H} \left(\bigvee_{x \in h} f(x) \vee \bigvee_{\bar{x} \in h} \neg f(x) \right) = 1$$

を満たす f が存在するか？

3-PARTITION の定義は以下の通り．

インスタンス： 以下の条件を満たす，整数の集合 $S = \{a_1, \ldots, a_{3t}\}$ と 整数 b.

- $b/4 < a_i < b/2$
- $\displaystyle\sum_{i=1}^{3t} a_i = tb$

問題： 以下の条件を満たす $\mathcal{A} \subseteq 2^S$ が存在するか？

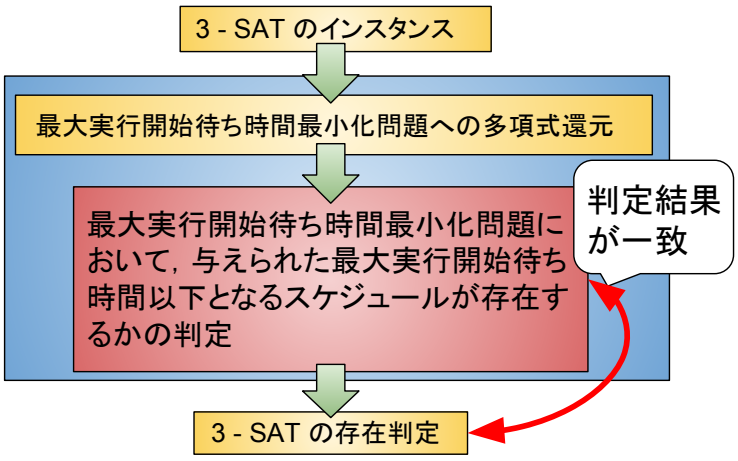
- $\forall A \in \mathcal{A}[|A| = 3 \wedge \sum_{a \in A} a = b]$
- $\bigcup_{A \in \mathcal{A}} A = S$
- $\forall A, A' \in \mathcal{A}[A \neq A' \Rightarrow A \cap A' = \emptyset]$

3-SAT と 3-PARTITION の計算複雑さは NP 完全であることが Garey and Johnson [1990] [2] により証明されている．ある決定問題が NP 完全であるとは，以下の二つが成立することを示す．

- その問題が NP に属す．
- その問題が NP 困難である．

つまり，すべての NP である問題から還元でき，加えて， Yes となる証拠が与えられた時その証拠が正しいことを多項式時間で確認できることを示す必要がある．ただし全ての NP である問題は推移性が成り立つためどれか 1 つから還元することが出来れば良い．

本研究では，3-SAT からの多項式時間還元により 無関連並列機械モデルにおいて，機械数が入力の一部の場合，最大実行開始待ち時間最小化問題が NP 完全であることを示す．つまり，3-SAT のインスタンスから 最大実行開始待ち時間最小化問題 のインスタンスが多項式時間で構成可能であり，加えて， 3-SAT の判定結果が Yes ならばかつその場合に限り構成された 最大実行開始待ち時間最小化問題のインスタンスに対し判定結果が Yes であることを示す．加えて，スケジュールを証拠とし，最大実行開始待ち時間最小化問題 のインスタンスとスケジュールが与えられたとき，そのスケジュールにおける最大実行開始待ち時間が w 以下を満たすかの判定が多項式時間で行えることを示すことにより，最大実行開始待ち時間最小化問題が NP 完全であることを示す．



2.2.2 既存のスケジューリング問題との対応

以下では，最大実行開始待ち時間最小化問題と，JIT ジョブ荷重和最大化問題と処理開始可能時刻付き最大遅れ時間最小化問題との共通部分と差分をまとめる．

- **JIT ジョブ荷重和最大化問題との共通部分**

  実行開始待ち時間 w の制約が最も強い場合，つまり，$w = 0$ のとき，各ジョブは処理開始可能時刻ちょうどで処理を開始しなければならない．このとき，最大実行開始待ち時間最小化問題は JIT ジョブスケジューリング問題として捉えることができる．つまり，最大実行開始待ち時間最小化問題は JIT ジョブ荷重和最大化問題の拡張問題として捉えることができる．

- **JIT ジョブ荷重和最大化問題との差分**

  JIT ジョブ荷重和最大化問題は，納期以前に処理を完了したジョブの荷重和最大化する問題である．対して，最大実行開始待ち時間最小化問題（決定問題）では，すべてのジョブの実行開始待ち時間が w 以下となるスケジュールの存在判定問題である．つまり，JIT ジョブ荷重和最大化問題では，目的関数に影響を与えるジョブが全体の一部であることに対し，最大実行開始待ち時間最小化問題では，全てのジョブが影響を与える．

- **処理開始可能時刻付き最大遅れ時間最小化問題との共通部分**

  最大実行開始待ち時間最小化問題（決定問題）は，ジョブが処理開始可能時刻と（処理開始可能時刻 $+$ 処理時間 $+ w$）の 区間で処理可能かという問題として捉えることができる．また，処理開始可能時刻付き最大遅れ時間最小化問題を最大遅れ時間が ℓ 以下となるスケジュールが存在するか，という決定問題として定義したとき，ジョブは処理開始可能時刻と（納期 $+ \ell$）の区間で処理可能かという問題として捉えることができる．このとき，どちらの問題も区間スケジューリング問題として捉えることができる．

- **処理開始可能時刻付き最大遅れ時間最小化問題との差分**

  処理開始可能時刻付き最大遅れ時間最小化問題において，納期 $:=$ 処理開始可能時刻 $+$ 処理時間 としたとき，この問題は，処理開始可能時刻付き最大遅れ時間最小化問題の部分問題である．最大実行開始待ち時間最小化問題において，納期は，処

理開始可能時刻 + 処理時間 + w で表すことができる．つまり，最大実行開始待ち時間最小化問題は処理開始可能時刻付き最大遅れ時間最小化問題の部分問題として捉えることができる．

以上の点から，最大実行開始待ち時間最小化問題は，既存のスケジューリング問題の部分問題，もしくは拡張問題として定義できる．上記の共通部分，差分を考慮し，以下で，最大実行開始待ち時間最小化問題の計算複雑さの証明や解法の提案を行う．

第3章　従来研究

第 3 章では，従来研究の成果を紹介する．第 3.1 節では，JIT ジョブ荷重和最大化問題について，計算複雑さの観点から紹介する．第 3.2 節では，処理開始可能時刻付き最大遅れ時間最小化問題について，計算複雑さと解法の観点から紹介する．

3.1　JITジョブ荷重和最大化問題

この問題は　3-SAT からの還元により，強 NP 困難であることが Sung and Vlach [2005] [9] によって証明されている．以下では，JIT ジョブ荷重和最大化問題を定式化し，問題が NP 困難であることの証明を示す．

3.1.1　定式化

入力： n 個のジョブ $J_1, \ldots, J_n$ を m 台の同一機械 $M_1, \ldots, M_m$ で処理する．入力は，各ジョブ $J_i (i \in \{1, \ldots, n\})$ の，処理時間 p_i ，処理開始可能時刻 r_i ，納期 d_i ，荷重 w_i であり，それぞれの値は自然数である．各ジョブの処理時間をまとめて $P = (p_1, \ldots, p_n) \in \mathbb{N}^n$，処理開始可能時刻をまとめて $R = (r_1, \ldots, r_n) \in \mathbb{N}^n$ ，納期をまとめて $D = (d_1, \ldots, d_n) \in \mathbb{N}^n$ ，荷重をまとめて $W = (w_1, \ldots, w_n) \in \mathbb{N}^n$ と表し，入力 は 4 項組，(P, R, D, W) と表す．

解： 問題の前提に基づき，スケジュールを定式化する．スケジュールは以下の条件を満たす $\forall j \in \{1, \ldots, m\}\big[A_j \subseteq \{1, \ldots, n\}\big]$ と $C : \{1, \ldots, n\} \to \mathbb{N}$ の対 (A, C) であり，スケジュールによって，各ジョブを，どの機械で，いつ処理をするかを決める．

- $\forall i, i' \in \{1, \ldots, n\} \left[\big[i \neq i' \wedge A_i = A_{i'}\big] \Rightarrow [C(i) - p_j, C(i)) \cap [C(i') - p_{i'}, C(i')) = \emptyset\right]$
- $\forall i \in \{1, \ldots, n\}\big[C(i) - p_i \geq r_i\big]$

目的関数： 実行可能なスケジュール (A, C) のうち，納期以前に処理を完了するジョブの

荷重和，

$$\varphi(A,C) = \sum_{i \in \mathcal{Q}(A,C)} w_i$$

，$i \in \mathcal{Q}(A,C)$ を最大とするスケジュールを求める．ただし，$\mathcal{Q}(A,C)$ はスケジュール (A,C) において納期以前に処理を完了するジョブの集合である ($\mathcal{Q}(A,C) = \{i \in \{1,\ldots,n\} | C(i) \geq d_i\}$).

3.1.2 計算複雑さの証明

3-SAT からの還元により，JIT ジョブ荷重和最大化問題の計算複雑さの証明を行う．

ブール型変数の集合 $X = \{x_1,\ldots,,x_\beta\}$ と， X 上の 3 つのリテラルからなる集合 $H = h_1,\ldots,h_\lambda$ の 2 項組， (X,H) を任意の 3-SAT 問題の入力とする．議論に先立ち，いくつかの表記を導入する．各 $k \in \{1,\ldots,\beta\}$ について， γ_k を H において x_k が現れる回数を表す自然数とし， γ を $\gamma_k (k \in \{1,\ldots,\beta\})$ の最大値に 1 を加えた値とする $(\gamma = \max_{k \in \{1,2,\ldots,\beta\}} \{\gamma_k\} + 1)$. (X,H) に基づき，以下のように JIT ジョブ荷重和最大化問題の入力を設定する. $2\beta_\gamma$ 個のジョブを $\lambda + \beta$ 台の無関連機械で処理する. 各ジョブの処理時間，納期ズレ幅，納期，荷重を以下のように設定する.

納期ズレ幅の設定： 各ジョブ $i \in \{1,\ldots,2\beta\gamma\}$ の納期ズレ幅 α_i を 0 とする.

納期の設定： $\{1,2,\ldots,\beta\gamma\}$の$\beta\gamma$ 個ジョブの納期について，昇順で γ 個ずつとった β 個のグループについて，機械を順に対応させる．各グループ $k \in \{1,2,\ldots,\beta\}$ の γ 番目のジョブの納期を $2\gamma - 1$，ℓ 番目のジョブ ($\ell \neq \gamma$) の納期を ℓ とする．つまり，各 $k \in \{1,2,\ldots,\beta\}$ と $\ell \in \{1,2,\ldots,\gamma\}$ について，

$$d_{(k-1)\gamma+\ell} = \begin{cases} \ell & \text{if } \ell < \gamma \\ 2\gamma - 1 & \text{if } \ell = \gamma \end{cases}$$

$\{\beta\gamma + 1,\ldots,2\beta\gamma\}$ の $\beta\gamma$ 個のジョブの納期についても，同様に対応させる．各グループ $k \in \{1,2,\ldots,\beta\}$ の γ 番目のジョブの納期を γ，ℓ 番目のジョブ ($\ell \neq \gamma$) の納期を $\gamma + \ell$ とする．つまり，各 $k \in \{1,2,\ldots,\beta\}$ と $\ell \in \{1,2,\ldots,\gamma\}$ について，

$$d_{(\beta+k-1)\gamma+\ell} = \begin{cases} \gamma + \ell & \text{if } \ell < \gamma \\ \gamma & \text{if } \ell = \gamma \end{cases}$$

荷重和の設定： 各ジョブを納期昇順に基づき 2β 個のグループに分ける．グループ $k \in \{1,2,\ldots,\beta\}$ と，グループ $k + \beta$ について，グループの γ 番目のジョブの荷重を

γ とし，それ以外のジョブの荷重を 1 とする．つまり，各 $k \in \{1,2,\ldots,\beta\}$ と $\ell \in \{1,2,\ldots,\gamma\}$ について，

$$w_{(k-1)\gamma+\ell^v} = w_{(\beta+k-1)\gamma+\ell} = \begin{cases} 1 & \text{if } \ell < \gamma \\ \gamma & \text{if } \ell = \gamma \end{cases}$$

処理時間の設定： 各ジョブを納期昇順に基づき 2β 個のグループに分けるグループ $k \in \{1,2,\ldots,\beta\}$ に機械を k に対応させ，グループ $k+\beta$ にも同様に機械 k を対応させる．各グループの各ジョブについて， グループに対応した機械以外の各機械における処理時間を 2γ とする．各グループの γ 番目のジョブの，グループに対応する機械における処理時間を γ とし，γ 番目のジョブをのぞいた各ジョブの，グループに対応する機械における処理時間を 1 とする．つまり，各 $j \in \{1,2,\ldots,\beta\}$ と $k \in \{1,2,\ldots,\beta\}$ と $\ell \in \{1,2,\ldots,\gamma\}$ について，

$$p_{(k-1)\gamma+\ell,j} = p_{(\beta+k-1)\gamma+\ell,j} = \begin{cases} 1 & \text{if } \ell < \gamma \text{ and } j = k, \\ \gamma & \text{if } \ell = \gamma \text{ and } j = k, \\ 2\gamma & \text{otherwise} \end{cases}$$

残りの各機械 $j \in \{\beta+1,\ldots,\beta+\gamma\}$ について，各ジョブの処理時間を設定する各グループ $k \in \{1,2,\ldots,\beta\}$ について，x_k を対応させる．各グループ $k \in \{1,2,\ldots,\beta\}$ について，$\ell \in \{1,2,\ldots,\gamma-1\}$ 番目の各ジョブにつ いて，$x_k \in h_{j-\beta}$ であれば $d_{(k-1)\gamma+1}$ とし，そうでない場合は 2γ とする．また，γ 番目 のジョブの処理時間は 2γ とする. グループ $k+\beta$ について $\bar{x}_k$ を対応させる．$\ell \in \{1,2,\ldots,\gamma-1\}$ 番目の各ジョブについて，$\bar{x}_k \in h_{j-\beta}$ であれば $d_{(\beta+k-1)\gamma+\ell}$ とし， そうでない場合は 2γ とする.また，γ 番目のジョブの処理時間は 2γ とする．つまり，各 $j \in \{\beta+1,\ldots,\beta+\gamma\}$ と $k \in \{1,2,\ldots,\beta\}$ と $\ell \in \{1,2,\ldots,\gamma\}$ について，

$$p_{(k-1)\gamma+\ell,j} = \begin{cases} d_{(k-1)\gamma+1} & \text{if } \ell < \gamma \text{ and } x_k \in h_{j-\beta}, \\ 2\gamma & \text{otherwise} \end{cases}$$

$$p_{(\beta+k-1)\gamma+\ell,j} = \begin{cases} d_{(\beta+k-1)\gamma+\ell} & \text{if } \ell < \gamma \text{ and } \bar{x}_k \in h_{j-\beta}, \\ 2\gamma & \text{otherwise} \end{cases}$$

このように設定した (P,a,D,W) は JIT ジョブ荷重和最大化問題の入力である．以降，(X,H) に基づいて前述のように設定した (P,a,D,W) を区別して $I_{(X,H)}$ と表す ($I_{(X,H)} = (P,a,D,W)$)．以下，H を充足する真理値割り当て $f : X \to \{0,1\}$ が存在することと，$I_{(X,H)}$ を入力とする JIT ジョブ荷重和最大化問題に対して $\varphi(A,C) \geq \beta(2\gamma-1)+\lambda$ を満たす実行可能なスケジュール (A,C) が存在することが，同値であることを示す．

補題 1 H を充足する真理値割り当て $f : X \to \{0,1\}$ が存在するならば，$I_{(X,H)}$ を入力とする JIT ジョブ荷重和最大化問題に対して $\varphi(A,C) \geq \beta(2\gamma-1)+\lambda$ を満たす実行可能なスケジュール (A,C) が存在する.

証明 (X,H) を任意の 3 -SAT 問題の入力とし， $f : X \to \{0,1\}$ を H を充足する真理値割り当てとする. つまり， f は以下を満たす.

$$\bigwedge_{h\in H}\left(\bigvee_{x\in h} f(x) \vee \bigvee_{\bar{x}\in h} \neg f(x)\right) = 1$$

各 $k \in \{1,\ldots,\beta\}$ について， $f(x_k) = 1$ ならばグループ $\beta+k$ の全てのジョブを，グループ $\beta+k$ に対応する機械 k に割り当て，そうでなければグループ k の全てのジョ ブを，グループ k に対応する機械 k に割り当てる. また，機械 k に割り当てた各ジョ ブの完了時刻を納期とする. つまり，$f(x_k) = 1$ ならば，各 $\ell \in \{1,\ldots,\gamma\}$ について $A_k := A_k cup\{(\beta+k-1)\gamma+\ell\}$ かつ $C((\beta+k-1)\gamma+\ell) = d(\beta+k-1)\gamma+l$ ， $f(x_k) = 0$ ならば，各 $\ell \in \{1,\ldots,\gamma\}$ について $A_k := A_k \cup \{(k-1)\gamma+\ell\}$ かつ $C((k-1)\gamma+\ell) = d(k-1)\gamma+\ell$. $I_{(X,H)}$ の定義より，ここまでのスケジュールにおいて，機械 $1,\ldots,\beta$ に割り当てたジョ ブの処理時間は重複せず (図 3.1)，完了時刻は納期であることから，$k \in \{1,\ldots,\beta\}$ に割り当てたジョブは JIT ジョブである. よって，ここまでのスケジュール (A,C) における JIT ジョブの荷重和は，

$$\varphi(A,C) = \sum_{k\in\{1,\ldots,\beta\}}\left(\sum_{i\in\mathcal{Q}(A,C):i\in A_k} w_i\right) = \sum_{k\in\{1,\ldots,\beta\}}(2\gamma-1) = \beta(2\gamma-1)$$

.

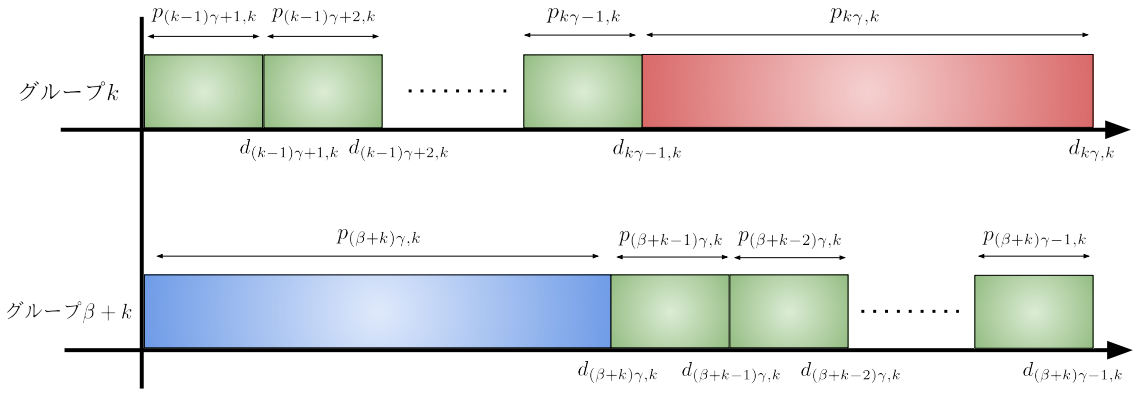


図 3.1: 機械 $k \in \{1,\ldots,\beta\}$ に対応するグループ k とグループ $\beta+k$ の各ジョブ

各 $j \in \{1,\ldots,\lambda\}$ について，$f(x_k) = 1$ を満たす $x_k \in h_j$ か，$f(x_k) = 0$ を満たす $\bar{x}_k \in h_j$ をひとつ見つける. f が H を充足することから，各 $j \in \{1,\ldots,\lambda\}$ について， 上記を

満たすリテラルは必ず存在する．与えられた $j \in \{1,\ldots,\lambda\}$ について， $f(x_k)=1$ を満たす $x_k \in h_j$ を見つけたと仮定す る．グループ k のジョブのうち，いずれの機械にも割り当てられていない $\ell \in \{1,\ldots,\gamma-1\}$ を見つける．グループ k のジョブは，$f(x_k)=1$ であることから，$\{1,\ldots,\beta\}$ のいずれの機械にも割り当てられておらず，また，各リテラルが現れる回数は高々$\gamma-1$ であることから，そのような ℓ 番目のジョブは必す存在する． $A_{j+\beta} := A_{j+\beta} \cup \{(k-1)\gamma+\ell\}$ とし， $C((k-1)\gamma+l) = d(k-1)\gamma+\ell$ とする．次に，与えられた $j \in \{1,\ldots,\lambda\}$ について， $f(x_k)=0$ を満たす $\bar{x}_k \in h_j$ を見つけたと仮定する．グループ $\beta+k$ のジョブのうち，いずれの機械にも割り当てられていない $\ell \in \{1,\ldots,\gamma\}$ を見つける．先ほどと同様の議論で，そのような ℓ は必ず存在する．$A_{j+\beta} := A_{j+\beta} \cup \{(\beta+k-1)\gamma+\ell\} = j+\beta$ とし，$C((\beta+k-1)\gamma+\ell) = d(\beta+k-1)\gamma+\ell$ とする．各グループの $\ell \in \{1,\ldots,\gamma-1\}$ 番目のジョブの荷重が 1 であることから，各 $j \in \{1,\ldots,\lambda\}$ について j に割り当てられた JIT ジョブの荷重和は，

$$\sum_{i \in \mathcal{Q}(A,C):A(i)=j+\beta} w_i = 1$$

ここまでのスケジュール (A,C) においていずれの機械にも割り当てられていないジョブを， $\max\{d_i | i \in \{1,\ldots,2\beta\gamma\}$ のあと，任意の順，任意の機械で処理するものとする．明らかにスケジュール (A,C) は実行可能であり，以下を満たす．

$$\varphi(A,C) = \sum_{j\in\{1,\ldots,\beta\}} \sum_{i\in\mathcal{Q}(A,C):i\in A_j} w_i + \sum_{j\in\{\beta+1,\ldots,\lambda+\beta\}} \sum_{i\in\mathcal{Q}(A,C):i\in A_j} w_i = \beta(2\gamma-1)+\lambda$$

．■

ここで， $I_{(X,H)}$ を入力とする JIT ジョブ荷重和最大化問題における実行可能なスケジュールについて，いくつかの性質を示す．(A,C) を $_{I(X,H)}$ を入力とする JIT ジョブ荷重和最大化問題における任意の実行可能なスケジュールとする．実行可能性の制約により，各ショブは機械の稼働開始時刻以降に処理を開始することから，各 JIT ジョブ $i \in \mathcal{Q}(A,C)$ について，$C(i)-p_i = d_i-p_i \geq 0$ である．このことから，$p_i, j \geq 2\gamma$ かつ $d_i \leq 2\gamma-1$ を満たす $i \in \{1,\ldots,n\}$ と $j \in \{1,\ldots,m\}$ について，$i \in \mathcal{Q}(A,C)$ のとき，$i \notin A_j$ である．したがって，$I_{(X,H)}$ の定義より，各 $k \in \{1,\ldots,\beta\}$ について，

- $i \in \mathcal{Q}(A,C)$ かつ $i \in A_k$ ならば，$(k-1)\gamma+1 \leq i \leq k\gamma$ または $(\beta+k-1)\gamma+1 \leq i \leq (\beta+k)\gamma$ である．

- $k\gamma \in \mathcal{Q}(A,C)$ ならば $k\gamma \in A_k$ であり，$\beta+k)\gamma \in \mathcal{Q}(A,C)$ ならば $(\beta+k)\gamma \in A_k$ である．

実行可能性の制約より，同じ機械に割り当てられた複数のジョブの処理は重複しないことから，各 $k \in \{1,\ldots,\beta\}$ について，

- $k\gamma \in \mathcal{Q}(A,C)$ ならば，$i \in A_k$ である任意の $i \in \mathcal{Q}(A,C)$ について，i は $(k-1)\gamma - 1 \leq k\gamma$ を満たす (図 3.1)．
- $(\beta+k)\gamma \in \mathcal{Q}(A,C)$ ならば，$i \in A_k$ てある任意の $i \in \mathcal{Q}(A,C)$ について，i は $(\beta+k-1)\gamma - 1 \leq i \leq (\beta+k)\gamma$ を満たす．

以上の議論より，各 $k \in \{1,\ldots,\beta\}$ について，k に割り当てられた JIT ジョブの荷重和は以下を満たし

$$\sum_{i\in\mathcal{Q}(A,C):i\in A_k} w_i \leq 2\gamma - 1 \tag{A.1}$$

以下のいずれかを満たす場合にのみ，$\sum_{i\in\mathcal{Q}(A,C):i\in A_k} w_i = 2\gamma - 1$ となる．

$$\{i \in \mathcal{Q}(A,C) \mid i \in A_k\} = \{(k-1)\gamma+1, (k-1)\gamma+2, \ldots, k\gamma\} \tag{A.2}$$

$$\{i \in \mathcal{Q}(A,C) \mid i \in A_k\} = \{(\beta+k-1)\gamma+1, (\beta+k-1)\gamma+2, \ldots, (\beta+k)\gamma\} \tag{A.3}$$

さらに，各 $j \in \{\beta+1,\ldots,\beta+\lambda\}$ について，各ジョブ $i \in \{1,\ldots,2\beta\gamma\}$ の処理時間は $p_ij \in \{d_i, 2\gamma\}$ であり，機械 j に割り当てられるジョブは高々 1 つである ($|\{i \in \mathcal{Q}(A,C)\}|i \in A_j\}| \leq 1$)．既に述べたように，各 $k \in \{1,\ldots,\beta\}$ について $k\gamma \in \mathcal{Q}(A,C)$ ならば $k\gamma \in A_k$ であり，$(\beta+k)\gamma \in \mathcal{Q}(A,C)$ ならば，$(\beta+k)\gamma \in A_k$ である．これらのジョブをのぞいた各ジョブの荷重は 1 であることから，各 $j \in \{\beta+1,\ldots,\lambda+\beta\}$ について，

$$\sum_{i\in\mathcal{Q}:i\in A_j} w_i \leq 1 \tag{A.4}$$

補題 2 *$I_{(X,H)}$ を入力とする JIT ジョブ荷重和最大化問題に対して $\varphi(A,C) \geq \beta(2\gamma-1)+\lambda$ を満たす実行可能なスケジュール (A,C) が存在するならば，H を充足する真理値割り当て $f : X \to \{0,1\}$ が存在する．*

証明 (A,C) を $I_{(X,H)}$ を入力とする JIT ジョブ荷重和最大化問題に対して $\varphi(A,C) \geq \beta(2\gamma-1)+\lambda$ を満たす実行可能なスケジュールとする．式 (15) と式 (A.4) より $\varphi(A,C) \leq$

$\beta(2\gamma-1)+\lambda$ であり，このことから，$\varphi(A,C) = \beta(2\gamma-1)+\lambda$ である．各機械 $k \in \{1,\ldots,\beta\}$ について，式 $(A.2)$ または式 $(A.3)$ が満たされ，各 $j \in \{\beta+1,\ldots,\beta+\lambda\}$ について，$|\{i \in \mathcal{Q}(A,C) | i \in A_j\}| = 1$ である．(A,C) に基づき，$f : H \to \{0,1\}$ を以下のように設定する．各 $k \in \{1,\ldots,\beta\}$ について，式 $(A.3)$ が満たされるならば $f(x_k) = 1$ とし，そうでなければ $f(x_k) = 0$ とする．明らかに，各 $j \in \{\beta+1,\ldots,\beta+\lambda\}$ について j に割り当てられたただ一つの JIT ジョブ $i \in \mathcal{Q}(A,C)$ が存在し，$i \geq \beta\gamma$ ならば $f(x_k) = 1$ を満たすある $x_k \in h_{j-\beta}$ が存在し，$i > \beta\gamma$ ならば $f(x_k) = 0$ を満たすある $\bar{x}_k \in h_{j-\beta}$ が存在する．したがって，f は H を充足する真理値割り当てである． ■

補題 1 と補題 2 より，H を充足する真理値割り当て $f : X \to \{0,1\}$ が存在することと，$I_{(X,H)}$ を入力とする JIT ジョブ荷重和最大化問題に対して $\varphi(A,C) \geq \beta(2\gamma-1)+\lambda$ を満たす実行可能なスケジュール (A,C) が存在することは，同値である．3 -SAT 問題の入力 (X,H) に基づく $I_{(X,H)}$ の生成に要する計算時間は，明らかに多項式時間であり，入力 $I_{(X,H)}$ の長さは，(X,H) の長さに関する多項式で表すことができる．したがって，無関連並列機械モデルにおいて機械数が入力の一部の場合， JIT ジョブ荷重和最大化問題は強 NP 困難である．

3.2 処理開始可能時刻付き最大遅れ時間最小化問題

この問題は 3-PARTITION からの還元により，強 NP 困難であることが Garey and Johnson [1977] によって，証明されている．[2] 以下では，処理開始可能時刻付き最大遅れ時間最小化問題を定式化し，問題が NP 困難であることの証明を示す．

3.2.1 定式化

入力： n 個のジョブ $J_1,\ldots,J_n$ を 1 台の機械で処理する．入力は，各ジョブ $J_i (i \in \{1,\ldots,n\})$ の，処理時間 p_i，処理開始可能時刻 r_i，納期 d_i であり，それぞれの値は自然数である．各ジョブの処理時間をまとめて $P = (p_1,\ldots,p_n) \in \mathbb{N}^n$ ，処理開始可能時刻をまとめて $R = (r_1,\ldots,r_n) \in \mathbb{N}^n$ ，納期をまとめて $D = (d_1,\ldots,d_n) \in \mathbb{N}^n$ と表し，入力は 3 項組 (P,R,D) と表す．

解： 問題の前提に基づき，スケジュールを定式化する．スケジュールは以下の条件を満たす $C : \{1,\ldots,n\} \to \mathbb{N}$ であり，スケジュールによって，各ジョブ，をいつ処理を

するかを決める.

- $\forall i, i' \in \{1, \ldots, n\} \left[\left[i \neq i' \right] \Rightarrow [C(i) - p_i, C(i)) \cap [C(i') - p_{i'}, C(i')) = \emptyset \right]$
  - 機械は同時に複数のジョブを処理しない
  - 各ジョブの処理を開始すると，完了するまで中断しない
- $\forall i \in \{1, \ldots, n\} \left[C(i) - p_i \geq r_i \right]$
  - 各ジョブは処理開始可能時刻以降によりを開始する

目的関数： 実行可能なスケジュール C のうち，最大の納期遅れ，

$$\varphi(C) = \max_{i \in \{1,\ldots,n\}} \{C(i) - d_i\}$$

を最小とするスケジュールを求める.

3.2.2 計算複雑さの証明

3-PARTITION からの還元により，処理開始可能時刻つき最大遅れ時間最小化問題の計算複雑さを証明する.

整数の集合 $S = \{a_1, \ldots, a_{3t}\}$ と 整数 b の 2 項組 (S, b) を 3-PARTITION の任意の入力とする．(S, a) に基づき，以下のように最大遅れ時間最小化問題の入力を設定する．$4t - 1$ 個のジョブを 1 台の機械で処理する．各ジョブの処理時間，納期ズレ幅，納期を以下のように設定する.

$\mathcal{A} \subseteq 2^S$ が以下を満たすとき，3-PARTITION の解は Yes.

- $\forall A \in \mathcal{A}[|A| = 3 \wedge \sum_{a \in A} a = b]$
- $\bigcup_{A \in \mathcal{A}} A = S$
- $\forall A, A' \in \mathcal{A}[A \neq A' \Rightarrow A \cap A' = \emptyset]$

処理時間開始可能時刻の設定： 各 $i \in \{1, \ldots, 4t - 1\}$ におけるジョブ J_i の処理開始可能時刻について，$1 \leq i \leq t-1$ のとき，$r_i = ib + (i-1)$，$t \leq i \leq 4t-1$ のとき，$r_i = 0$ とする.

$$r_i = \begin{cases} ib + (i-1) & \text{if } 1 \leq i \leq t-1, \\ 0 & \text{if } t \leq i \leq 4t-1 \end{cases}$$

処理時間の設定： 各 $i \in \{1, \ldots, 4t-1\}$ におけるジョブ J_i の処理時間について，$1 \leq i \leq t-1$ のとき，$p_i = 1$，$t \leq i \leq 4t-1$ のとき，$p_i = a_{i-t+1}$ とする．

$$p_i = \begin{cases} 1 & \text{if } 1 \leq i \leq t-1, \\ a_{i-t+1} & \text{if } t \leq i \leq 4t-1 \end{cases}$$

納期の設定： 各 $i \in \{1, \ldots, 4t-1\}$ におけるジョブ J_i の処理時間について，$1 \leq i \leq t-1$ のとき，$d_i = ib + i$，$t \leq i \leq 4t-1$ のとき，$d_i = tb + (t-1)$ とする．

$$d_i = \begin{cases} ib + i & \text{if } 1 \leq i \leq t-1, \\ tb + (t-1) & \text{if } t \leq i \leq 4t-1 \end{cases}$$

このように設定した (P, R, D) は最大遅れ時間最小化問題の入力である．以降，(S, b) に基づいて前述のように設定した (P, R, D) を区別して $I_{(S,b)}$ と表す．以下，上記の条件を満たす $\mathcal{A} \subseteq 2^S$ が存在することと，$I_{(S,b)}$ を入力とする最大遅れ時間最小化問題に対して $\varphi(C) = 0$ を満たす実行可能なスケジュール C が存在することは，同値であることを示す．

補題 3 上記の条件を満たす $\mathcal{A} \subseteq 2^S$ が存在するならば，$I_{(S,b)}$ を入力とする最大遅れ時間最小化問題に対して $\varphi(C) = 0$ を満たす実行可能なスケジュール C が存在する．

証明 各 $1 \leq i \leq t-1$ におけるジョブ J_i を機械に割り当てる．各 $1 \leq i \leq t-1$ におけるジョブ J_i の処理時間は 1 であることから，$[0, tb + (t-1)]$ の区間は，$t-1$ 個のジョブによって，t 個区間に分断され，それぞれの長さは b である．(図 3.2)

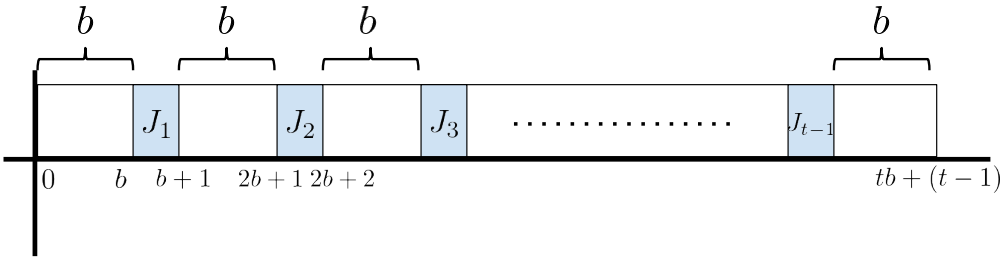


図 3.2: 各 $1 \leq i \leq t-1$ におけるジョブ J_i の配置による区間の分割

ここで，上記の条件を満たす $\mathcal{A} \subseteq 2^S$ が存在することから，$\forall A \in \mathcal{A}\big[|A| = 3 \wedge \sum_{a \in A} a = b\big]$ である．このとき，各 $t \leq i \leq 4t-1$ におけるジョブ J_i は 長さ b の t 個の区間のいずれかに割り当てられる．また，各区間に割り当てられた 3 つのジョブの処理時間の和は $I_{(S,b)}$ より，b である．

以上より，上記の条件を満たす $\mathcal{A} \subseteq 2^S$ が存在するならば，$I_{(S,b)}$ を入力とする最大遅れ時間最小化問題に対して $\varphi(C) = 0$ を満たす実行可能なスケジュール C が存在する．■

補題 4 上記の条件を満たす $\mathcal{A} \subseteq 2^S$ が存在しないならば，$I_{(S,b)}$ を入力とする最大遅れ時間最小化問題に対して $\varphi(C) \leq 0$ を満たす実行可能なスケジュール C が存在しない．

証明 各 $1 \leq i \leq t-1$ におけるジョブ J_i を機械に割り当てる．各 $1 \leq i \leq t-1$ におけるジョブ J_i の処理時間は 1 であることから，$[0, tb+(t-1)]$ の区間は，$t-1$ 個のジョブによって，t 個区間に分断され，それぞれの長さは b である．(図 3.2)

ここで，上記の条件を満たす $\mathcal{A} \subseteq 2^S$ が存在しないことから，$\forall A \in \mathcal{A}\big[|A| = 3 \wedge \sum_{a \in A} a \neq b\big]$ である．これより，各 $t \leq i \leq 4t-1$ におけるジョブ J_i は長さ b の t 個の区間のいずれかに割り当てるとき，各区間のジョブの処理時間の和が b でない区間が少なくとも 1 つ存在する．以上より，上記の条件を満たす $\mathcal{A} \subseteq 2^S$ が存在しないならば，$I_{(S,b)}$ を入力とする最大遅れ時間最小化問題に対して $\varphi(C) = 0$ を満たす実行可能なスケジュール C は存在しない．■

補題 3 と補題 4 より，上記の条件を満たす $\mathcal{A} \subseteq 2^S$ が存在することと，$I_{(S,b)}$ を入力とする最大遅れ時間最小化問題に対して $\varphi(C) \leq 0$ を満たす実行可能なスケジュール C が存在することは，同値である．3 -PARTITION 問題の入力 (S, b) に基づく $I_{(S,b)}$ の生成に要する計算時間は，明らかに多項式時間であり，入力 $I_{(S,b)}$ の長さは，(S, b) の長さに関する多項式で表すことができる．したがって，単一機械モデルにおいて，最大遅れ時間最小化問題は強 NP 困難である．

3.2.3 解法のまとめ

処理開始可能時刻付き最大遅れ時間最小化問題は，各制約により制限を加えることで多項式時間で最適解が求まることが Lawler [1973] [7]，Lageweg, Lenstra, and Rinnooy Kan [1976] [5] によって証明されている．つまり，処理開始可能時刻付き最大遅れ時間最小化問題の部分問題に対して，多項式アルゴリズムが存在する．各部分問題に対する解法は以下の通り．

処理開始可能時刻の制約に制限を加えた場合

つまり，$\forall j \in \{1,\ldots,n\}\big[r_j = r\big]$ を満たす．ジョブを納期の昇順で処理する (EDD ルール) ．つまり，スケジュールにおける任意の 2 つのジョブは以下の条件を満たす．

$$\forall i,j \in \{1,\ldots,n\}\big[d_i \leq d_j \Rightarrow C(i) \leq C(j)\big]$$

処理時間の制約により制限を加えた場合

つまり，$\forall j \in \{1,\ldots,n\}\big[p_j = p\big]$ を満たす．スケジュール可能なジョブの中で，納期が最も小さいジョブ順に処理する．つまり，スケジュールにおける各ジョブを順に $J_1,\ldots,J_n$，$C(0)=0$ とすると，各ジョブは以下の条件を満たす．

$$\forall j,j' \in \Big\{i \mid \forall i \in \{1,\ldots,n\}\big[r_i \leq C(i-1)\big]\Big\}\Big[j \leq j' \Rightarrow d_j \leq d'_j\Big]$$

納期の制約に制限を加えた場合

つまり，$\forall j \in \{1,\ldots,n\}\big[d_j = d\big]$ を満たす．ジョブを処理開始可能時刻の昇順で処理する．つまり，スケジュールにおける任意の 2 つのジョブは以下の条件を満たす．

$$\forall i,j \in \{1,\ldots,n\}\big[r_i \leq r_j \Rightarrow C(i) \leq C(j)\big]$$

ここで，処理開始可能時刻を一定とした最大遅れ時間最小化問題の部分問題に対して，EDD ルールを適用することで最適解が求まることの証明を示す．

補題 5 *処理開始可能時刻を一定とした最大遅れ時間最小化問題の部分問題に対して，EDD ルールを適用することで最適解が求まる．*

証明 スケジュール S をある 2 つの連続するジョブ J_i, J_k について納期順に並んでいないジョブを含むスケジュールとし，S' を納期の昇順で並んでいるスケジュールとする．ただし，全てのジョブの処理時間は 0 以上とする．つまり，スケジュール S におけるジョブ J_i，J_k の納期は，$d_i > d_k$ であり，スケジュール S' におけるジョブ J_i，J_k の納期は，$d_i < d_k$ である．上記の条件をまとめた図は以下の通りである．(図 3.3)

前提より，すべてのジョブの処理開始可能時刻が等しいことから，スケジュール S における，ジョブ J_i の処理開始時刻とスケジュール S' におけるジョブ J_k の処理開始時刻は等しい．つまり，$C_i - p_i = C'_k - p_k$ である．同様に，スケジュール S における，ジョブ J_k の完了時刻とスケジュール S' におけるジョブ J_i の完了時刻は等しい．つまり，$C_k = C'_i$ である．以上より，スケジュール S におけるジョブ J_i の遅れ時間 $L_i(S)$ は，$L_i(S) = C_i - d_i$，ジョブ J_k の遅れ時間 $L_k(S)$ は $L_k(S) = C_k - d_k$ となる．同様に，スケジュール S' におけるジョブ J_i の遅れ時間 $L_i(S')$ は，$L_i(S') = C'_i - d_i = C_k - d_i \leq C_k - d_k = L_k(S)$，

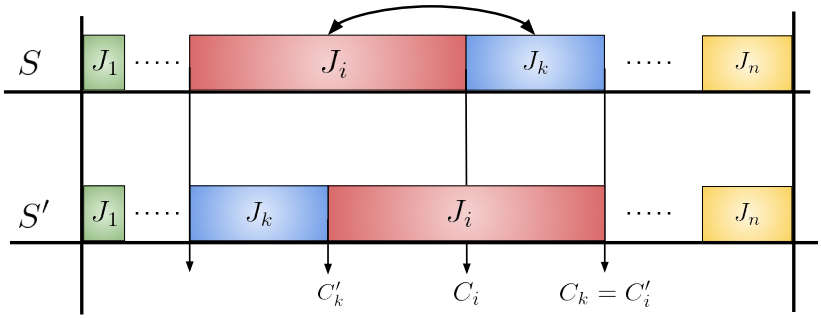


図 3.3: 納期順に割り当てられていないスケジュール S と納期順に割り当てられたスケジュール S'

ジョブ J_k の遅れ時間 $L_k(S')$ は，$L_k(S') = C'_k - d_k \leq C_k - d_k = L_k(S)$ と表すことができる．ここで，ジョブ J_i, J_k を除いたジョブにおける最大遅れ時間を $L(S)$ とする．つまり，$L(S) = \max_{j \in \{1,\ldots,n\} \setminus \{i,k\}} L_j(S)$．よって，スケジュール S，スケジュール S' における最大遅れ時間はそれぞれ，次のように表すことができる．

$$\begin{cases} L_{\max}(S) = \max\left\{L(S), L_i(S), L_k(S)\right\} \\ L_{\max}(S') = \max\left\{L(S), L_i(S'), L_k(S')\right\} \end{cases}$$

今までの議論より，$L_{\max}(S') \leq L_{\max}(S)$ であることは明らかである．

ここで，スケジュールの全体集合を $\mathcal{S}$，EDD ルールを適用したスケジュールの集合を $\mathcal{S}'$ とすると，前述の議論より，$\forall S \in \mathcal{S}$, $\exists S' \in \mathcal{S}'$, $L_{\max}(S') \leq L_{\max}(S)$ と表すことができる．また，どのスケジュールに対しても，EDD ルールを適用することで，一つの解に収束するので，$\forall S, S' \in \mathcal{S}'$, $L_{\max}(S) = L_{\max}(S')$ と表すことできる．以上より，$\forall S \in \mathcal{S}$, $\forall S' \in \mathcal{S}'$, $L_{\max}(S') \leq L_{\max}(S)$ となる． ■

補題 5 より，処理開始可能時刻を一定とした最大遅れ時間最小化問題の部分問題に対して，EDD ルールを適用することで最適解が求まることを示せた．

全てのジョブにおける処理時間や納期が常に一定であるときも，同様に証明できる．この解法は，本研究で扱う最大実行開始待ち時間最小化問題の部分問題の解法に適用でき最適解を求めることができる．

第4章　最大実行開始待ち時間最小化問題の計算複雑さ

最大実行開始待ち時間最小化問題は，どの機械モデルにおいても，計算複雑さが明らかでない．第 4.1 節では，無関連並列機械モデルにおける最大実行開始待ち時間最小化問題の NP 完全性の証明を行う．第 4.2 節では，最大実行開始待ち時間最小化問題の計算複雑さについてまとめる．

4.1　無関連並列機械モデルにおける NP 完全性の証明

ブール型変数の集合 $X = \{x_1, x_2, \ldots, x_n\}$ と， X 上の 3 つのリテラルからなる集合の集合 $H = \{h_1, h_2, \ldots, h_\lambda\}$ の 2 項組，(X, H) を任意の 3-SAT 問題の入力とする．

各 $i \in \{1, 2, \ldots, n\}$ について，α_i を H に おいて x_i が現れる回数を表す自然数とし，β_i を H において $\bar{x}_i$ が現れる回数を表す自然数とする．つまり，$\alpha_i = \big|\{h \in H \mid x_i \in h\}\big|$, $\beta_i = \big|\{h \in H \mid \bar{x}_i \in h\}\big|$ また， $\sum_{i \in \{1,\ldots,n\}} \alpha_i = \mathcal{A}$, $\sum_{i \in \{1,\ldots,n\}} \beta_i = \mathcal{B}$ とする．

ただし，各 $i \in \{1, 2, \ldots, n\}$ について，α_i, β_i は 3 以上とする．

つまり，$\forall i \in \{1, \ldots, n\}\big[\alpha_i \geq 3\big]$，$\forall i \in \{1, \ldots, n\}\big[\beta_i \geq 3\big]$

$(X.H)$ に基づき，以下のように 最大実行開始待ち時間最小化問題の入力を設定する．$2(\mathcal{A} + \mathcal{B}) + \lambda$ 個のジョブ $\mathcal{J}$ を $n + \lambda$ 台の無関連機械 $\mathcal{M}$ で処理する．

- $\mathcal{J}^t \subset \mathcal{J}$ s.t. $|\mathcal{J}^t| = \mathcal{A} + \mathcal{B}$,
- $\mathcal{J}^f \subset \mathcal{J}$ s.t.$|\mathcal{J}^f| = \mathcal{A} + \mathcal{B}$,
- $\mathcal{J}^d \subset \mathcal{J}$ s.t.$|\mathcal{J}^d| = \lambda$

ただし，$\mathcal{J} = \mathcal{J}^t \cup \mathcal{J}^f \cup \mathcal{J}^d$ ，$\mathcal{J}^t \cap \mathcal{J}^f = \emptyset$，$\mathcal{J}^f \cap \mathcal{J}^d = \emptyset$，$\mathcal{J}^d \cap \mathcal{J}^t = \emptyset$

つまり，$\{\mathcal{J}^t, \mathcal{J}^f, \mathcal{J}^d\}$ は $\mathcal{J}$ の分割である．

実行開始待ち時間の設定：

$$w = 2$$

まず，ジョブ $\mathcal{J}^d = \{J_1^d, \ldots, J_\lambda^d\}$ について，処理開始可能時刻関数 r を以下のように設定する．

$$r(\mathcal{J}^d) = 3(\mathcal{A} + \mathcal{B})$$

処理時間関数 p を以下のように設定する．

$$p(\mathcal{J}^d, A(\mathcal{J}^d)) = 3$$

次に，$2(\mathcal{A}+\mathcal{B})$ 個のジョブ $\mathcal{J}^t = \{J_1^t, \ldots, J_{\mathcal{A}+\mathcal{B}}^t\}$ と $\mathcal{J}^f = \{J_1^f, \ldots, J_{\mathcal{A}+\mathcal{B}}^f\}$ の処理開始可能時刻関数，処理時間関数を以下のように設定する．

処理開始可能時刻関数の設定： 各機械 $i \in \{1, 2, \ldots, n\}$ に対応させた各ジョブの処理時間関数を設定する．各ジョブを添え字の昇順に基づき $2n$ 個のグループに分ける． 各 $i \in \{1, 2, \ldots, n\}$ における i 番目のジョブの集合を $\mathcal{J}^t(i)$ とし，$n+i$ 番目のジョブの集合を $\mathcal{J}^f(i)$ とする．また，グループ i の ℓ 番目のジョブを $J^t(i, \ell)$ または $J^t(i, \ell)$ と表記する．

- ジョブの集合 $\mathcal{J}^t$ について， 各グループ $i \in \{1, 2, \ldots, n\}$ の ℓ 番目のジョブの処理開始可能時刻関数を各 i における $\ell \in \{1, 2, \ldots, \alpha_i + \beta_i\}$ について，以下のように設定する．

$$r(J^t(i,\ell)) = \begin{cases} 3 \displaystyle\sum_{j \in \{1,\ldots,i-1\}} (\alpha_j + \beta_j) + \ell - 1 & \text{if } \ell < \beta_i + 1, \\ 3 \displaystyle\sum_{j \in \{1,\ldots,i-1\}} (\alpha_j + \beta_j) + \alpha_i + \ell - 2 & \text{otherwise} \end{cases}$$

- ジョブの集合 $\mathcal{J}^f$ について，各グループ $i \in \{1, 2, \ldots, n\}$ の ℓ 番目のジョブの処理開始可能時刻を各 i における $\ell \in \{1, 2, \ldots, \alpha_i + \beta_i\}$ について，以下のように設定する．

$$r(J^f(i,\ell)) = \begin{cases} 3 \displaystyle\sum_{j \in \{1,\ldots,i-1\}} (\alpha_j + \beta_j) + \ell - 1 & \text{if } \ell = 1, \\ 3 \displaystyle\sum_{j \in \{1,\ldots,i-1\}} (\alpha_j + \beta_j) + \alpha_i + \ell & \text{else if } \ell < \alpha_i + 1, \\ 3 \displaystyle\sum_{j \in \{1,\ldots,i-1\}} (\alpha_j + \beta_j) + (\alpha_i + \beta_i + 4) + \ell & \text{otherwise} \end{cases}$$

処理時間関数の設定： 各ジョブを処理開始可能時刻の昇順に基づき $2n$ 個のグループに分ける

- グループ $i \in \{1,2,\ldots,n\}$ に機械 i に対応させ，グループ $n+i$ にも同様に機械 i を対応させる．各 $i \in \{1,2,\ldots,n\}$，$\ell \in \{1,2,\ldots,\alpha_i+\beta_i\}$ について，以下のように設定する．ただし，機械 i に対応しないジョブ，つまり，$A(\mathcal{J}^t) \neq i$ のとき，ジョブ $J^t \in \mathcal{J}^t$ の処理時間は $3(\mathcal{A}+\mathcal{B})+3$ とする．

$$p(J^t(i,\ell),A(\mathcal{J}^t)) = \begin{cases} 1 & \text{if } \ell < \beta_i, \\ \alpha_i+4 & \text{if } \ell = \beta_i, \\ 1 & \text{if } \ell < \alpha_i+\beta_i, \\ 3(\mathcal{A}+\mathcal{B}) - \Big\{3 \displaystyle\sum_{j\in\{1,\ldots,i-1\}} (\alpha_j+\beta_j) + (2\alpha_i+\beta_i-2)\Big\} + 3 & \text{if } \ell = \alpha_i+\beta_i \end{cases}$$

$$p(J^f(i,\ell),A(\mathcal{J}^f)) = \begin{cases} \beta_i+2 & \text{if } \ell = 1, \\ 1 & \text{if } \ell < \alpha_i, \\ \alpha_i+5 & \text{if } \ell = \alpha_i, \\ 1 & \text{if } \ell < \alpha_i+\beta_i, \\ 3(\mathcal{A}+\mathcal{B}) - \Big\{3 \displaystyle\sum_{j\in\{1,\ldots,i-1\}} (\alpha_j+\beta_j) + (2\alpha_i+2\beta_i+4)\Big\} + 3 & \text{if } \ell = \alpha_i+\beta_i \end{cases}$$

- 残りの各機械 $k \in \{n+1,\ldots,n+\lambda\}$ について，各ジョブの処理時間を設定する各グループ $i \in \{1,2,\ldots,n\}$ について x_i を対応させ，グループ $n+i$ について $\bar{x}_i$ を対応させる．つまり，$\forall i \in \{1,\ldots,n\}\big[x_i \to \mathcal{J}^t(i),\ \bar{x}_i \to \mathcal{J}^f(i)\big]$，$\mathcal{J}^t(i) = \big\{J^t(i,1),\ldots,J^t(i,\alpha_i+\beta_i)\big\}$，$\mathcal{J}^f(i) = \big\{J^f(i,1),\ldots,J^f(i,\alpha_i+\beta_i)\big\}$．各 $k \in \{n+1,\ldots,n+\lambda\}$ について，h_{k-n} に含まれるリテラルに対応するジョブを処理開始可能時刻の昇順で $\mathcal{T}(h_{k-n})$, $\mathcal{F}(h_{k-n})$ に入れる．$x_i \in h_{k-n}$ のとき，グループ $\mathcal{J}^t(i)$ のジョブを β_i+1 番目から順に $\mathcal{T}(k-n)$ に，$\mathcal{J}^f(i)$ のジョブを 1 番目から順に $\mathcal{F}(k-n)$ に加える．また，$\bar{x}_i \in h_{k-n}$ のとき，グループ $\mathcal{J}^f(i)$ のジョブを α_i+1 番目から順に $\mathcal{T}(k-n)$ に，$\mathcal{J}^t(i)$ のジョブを 1 番目から順に $\mathcal{F}(k-n)$ に加える．つまり，各 $k \in \{n+1,\ldots,n+\lambda\}$，各 $i \in \{1,\ldots,n\}$ について，$count(x_i,h_j) = \big|\big\{x_i \in h_l \mid l \in \{1,\ldots,j-1\}\big\}\big|$ を用いて，

$$\mathcal{T}(h_{k-n}) = \Big\{\big\{J^t(i,\beta_i+1+count(x_i,h_{k-n})) \mid x_i \in h_{k-n}\big\} \cup \big\{J^f(i,\alpha_i+1+count(\bar{x}_i,h_{k-n})) \mid \bar{x}_i \in h_{k-n}\big\}\Big\}$$

$$\mathcal{F}(h_{k-n}) = \Big\{\big\{J^f(i,1+count(x_i,h_{k-n})) \mid x_i \in h_{k-n}\big\} \cup \big\{J^t(i,1+count(\bar{x}_i,h_{k-n})) \mid \bar{x}_i \in h_{k-n}\big\}\Big\} \cup J^d_{k-n}$$

このとき，$i \in \{1,2,\ldots,n\}$，$k \in \{n+1,\ldots,n+\lambda\}$，$\ell \in \{1,2,\ldots,\alpha_i+\beta_i\}$ について，処理時間関数を次のように設定する．

$$p(J, A(J)) = \begin{cases} \min\{r(J') \mid J' \in \mathcal{F}(h_{k-n}) \wedge (r(J') > r(J))\} - r(J) & \text{if } J \in \mathcal{T}(h_{k-n}), \\ \min\{r(J') \mid J' \in \mathcal{F}(h_{k-n}) \wedge (r(J') > r(J))\} - r(J) + (4 - |h_j|) & \text{if } J \in \mathcal{F}(h_{k-n}), \\ 3(\mathcal{A} + \mathcal{B}) + 3 & \text{otherwise} \end{cases} \tag{B.1}$$

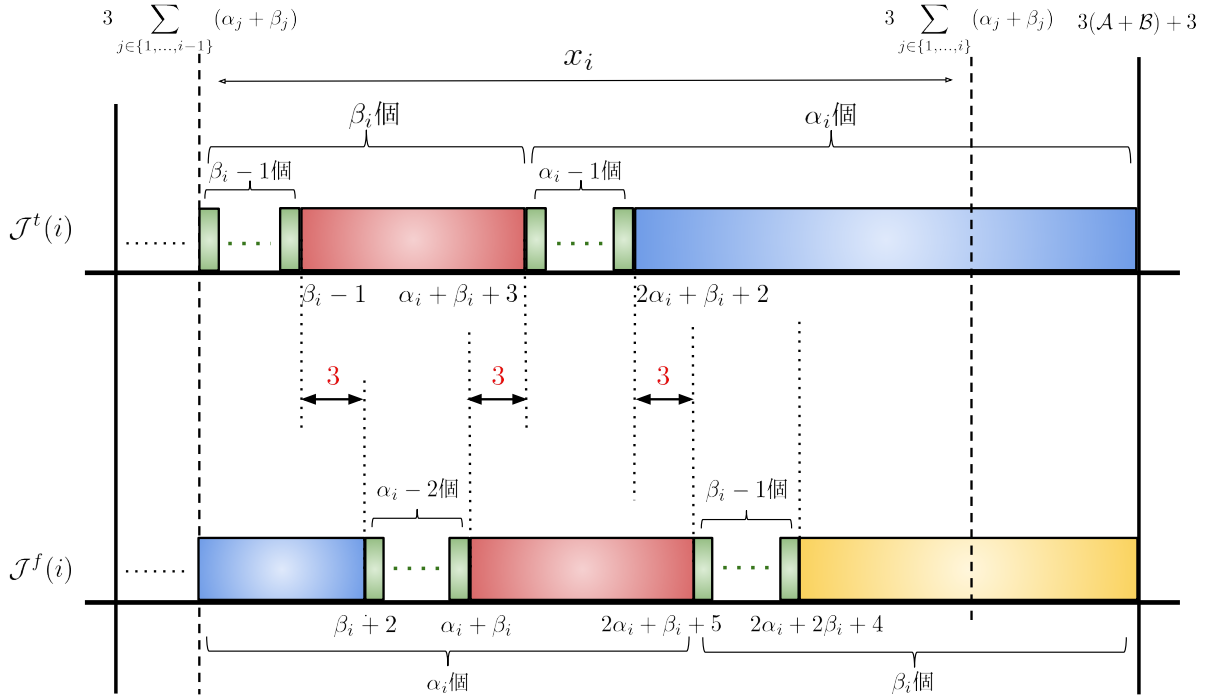


図 4.1: 各 $i \in \{1, \ldots, n\}$ における機械 i に対応するジョブの集合 $\mathcal{J}^t(i)$ または $\mathcal{J}^f(i)$ を機械 i に割り当てたときのスケジュール

このように設定した $(\mathcal{J}, \mathcal{M}, r, p, w)$ は最大実行開始待ち時間最小化問題の入力である．以降，(X, H) に基いて前述のように設定した $(\mathcal{J}, \mathcal{M}, r, p, w)$ を区別して $I_{(X,H)}$ と表す．つまり，$I_{(X,H)} = (\mathcal{J}, \mathcal{M}, r, p, w)$ ．

以下， H を充足する真理値割り当て $f : X \to \{0, 1\}$ が存在することと，$I_{(X,H)}$ を入力とする最大実行開始待ち時間最小化問題に対して，$\varphi(A, s) \leq 2$ を満たす実行可能なスケジュール (A, s) が存在することが，同値であることを示す．

補題 6 H を充足する真理値割り当て $f : X \to \{0, 1\}$ が存在するならば，$I_{(X,H)}$ を入力とする最大実行開始待ち時間最小化問題に対して $\varphi(A, s) \leq 2$ を満たす実行可能なスケジュール (A, s) が存在する．

証明 各 $i \in \{1, \ldots, n\}$ について，$f(x_i) = 1$ ならばグループ $n + i$ の全てのジョブを，グループ $n + i$ に対応する機械 i に割り当て，そうでなければグループ i の全てのジョ

ブを，グループ i に対応する機械 i に割り当てる．つまり，各 $i \in \{1,\ldots,n\}$ における各 $\ell \in \{1,\ldots,\alpha_i+\beta_i\}$ について，$f(x_i)=1$ ならば，$A_i := A_i \cup \mathcal{J}^f(i)$，$f(x_i)=0$ ならば，$A_k := A_k \cup \mathcal{J}^t(i)$ となる．$I_{(X,H)}$ の定義より，ここまでのスケジュールにおいて，機械 $\{1,\ldots,n\}$ に割り当てたジョブは処理開始可能時刻と納期の間で重複せず処理される．よって，このとき，機械 $\{1,2,\ldots,n\}$ に割り当てられた $\mathcal{A}+\mathcal{B}$ 個のジョブの実行開始遅れ時間は 0 である．(図 4.1)

上述の割り当てにより，各 $i \in \{1,\ldots,n\}$ について，x_i または $\bar{x}_i$ に対応するジョブ $\mathcal{J}^t(i)$ または $\mathcal{J}^f(i)$ のどちらか一方が機械 i に割り当てられた．このとき，上図より，機械 i におけるジョブの完了時刻は $3(\mathcal{A}+\mathcal{B})+3$ であるので，いずれのジョブも機械 i に割り当てることができない．よって，残りのジョブについては，各 $k \in \{n+1,\ldots,n+\lambda\}$ における機械 k に割り当てる．

前提条件より，f が H を充足することから，各 $i \in \{1,\ldots,n\}$，$j \in \{1,\ldots,\lambda\}$ について，$f(x_i)=1$ を満たす $x_i \in h_j$ か，$f(x_i)=0$ を満たす $\bar{x}_i \in h_j$ を満たすリテラルは必ず存在する．ここで，与えられた $j \in \{1,\ldots,\lambda\}$ について，$f(x_i)=1$ を満たす $x_i \in h_j$ を見つけたと仮定する．グループ i のジョブの集合 $\mathcal{J}^t(i)$ は，$f(x_i)=1$ であることから，$\{1,\ldots,n\}$ のいずれの機械にも割り当てられていないので，そのようなグループ i のジョブの集合 $\mathcal{J}^t(i)$ は必ず存在する．同様に，$f(x_i)=0$ を満たす $\bar{x}_i \in h_j$ を見つけたと仮定する．グループ $n+i$ のジョブの集合 $\mathcal{J}^f(i)$ のうち，いずれの機械にも割り当てられていないグループ $n+i$ を見つける．先程と同じ議論で，そのようなグループ $n+i$ のジョブ $\mathcal{J}^f(i)$ は必ず存在する．

各 $j \in \{1,\ldots,\lambda\}$ における h_j に対応するジョブの集合 $\mathcal{T}(h_j)$ または $\mathcal{F}(h_j)$ のジョブを処理開始可能時刻の昇順で機械 j に割り当てる．このとき，1 に割り当てられたリテラルに対応するジョブが少なくとも 1 つ各機械に割り当てられているので，各機械におけるジョブの完了時刻は高々$3(\mathcal{A}+\mathcal{B})+2$ である．ここで，各 $j \in \{1,\ldots,\lambda\}$ における J_j^d を各機械 j に 1 つずつ割り当てる．このとき，J^d の処理開始可能時刻は $I_{(X,H)}$ より，$3(\mathcal{A}+\mathcal{B})$ であるので，実行開始待ち時間は高々 2 である．

今までの議論より，$\mathcal{A}+\mathcal{B}$ 個のジョブを機械 $i \in \{1,\ldots,n\}$ に，$\mathcal{A}+\mathcal{B}$ 個のジョブを機械 $k \in \{n+1,\ldots,n+\lambda\}$ に，λ 個のジョブを機械 $k \in \{n+1,\ldots,n+\lambda\}$ に割り当てた．このとき，全てのジョブをいずれかの機械に割り当てており，全てのジョブに関して，実行開始待ち時間は高々 2 であることから，f が H を充足するとき，$I_{(X,H)}$ において，$\varphi(A,s) \leq 2$ を満たす． ■

補題 7 H を充足する真理値割り当て $f : X \to \{0,1\}$ が存在しないならば，$I_{(X,H)}$ を入力とする最大実行開始待ち時間最小化問題に対して $\varphi(A,C) \leq 2$ を満たす実行可能なスケジュール (A,s) が存在しない.

証明 f が H を充足しないことから，各 $j \in \{1,\ldots,\lambda\}$ について，$f(x_i) = 1$ を満たす $x_i \in h_j$ か，$f(x_i) = 0$ を満たす $\bar{x}_i \in h_j$ が 1 つも存在しない機械 $j \in \{1,\ldots,\lambda\}$ が必ず存在する．その機械を j' とし，以下，機械 j' 上のスケジュールについて述べる．

- $x_i \in h_{j'}$ または $\bar{x}_i \in h_{j'}$ に対応するジョブのみを機械 j' に割り当てた場合 $I_{(X,H)}$ の $B.3$ より，それらのジョブは，スケジュールにおける次のジョブの処理開始可能時刻を 1 超える処理時間を持つ．$h_{j'}$ に含まれる x_i 以外の残りの 2 つのリテラルに対応するジョブについても同様に機械 j' に割り当てる．$I_{(X,H)}$ より，機械 j' におけるジョブの完了時刻は $3(\mathcal{A}+\mathcal{B})+3$ となる．ここで，$j \in \{1,\ldots,\lambda\}$ におけるジョブ $J^d_{j'}$ をそれらのジョブの後に割り当てるとき，ジョブ $J^d_{j'}$ に 3 の実行開始待ち時間が発生する．$|h| = 2, |h| = 1$ のとき，$B.1$ より，それぞれの場合において $J^d \in \mathcal{J}^d$ に 4 または 3 の待ち時間が発生する.（図 4.2）

- $x_k \notin h_j$ または $\bar{x}_k \notin h_j$ に対応するジョブを割り当てた場合

  機械 j' にそのようなジョブを 1 つでも割り当てた場合，$I_{(X,H)}$ より，機械 j' におけるジョブの完了時刻は $3(\mathcal{A}+\mathcal{B})+3$ となる．ここで，$j \in \{1,\ldots,\lambda\}$ において，機械に割り当てられていないジョブ $J^d_{j'}$ をそれらのジョブの後に割り当てるとき，ジョブ $J^d_{j'}$ に 3 の実行開始待ち時間が発生する．

- 上記の 2 つのケースにおいて，$j \in \{1,\ldots,\lambda\}$ において，機械に割り当てられていないジョブ $J^d_{j'}$ を機械 $i \in \{1,\ldots,n\}$ に割り当てた場合 $I_{(X,H)}$ より，ジョブ $J^d_{j'}$ に 3 の実行開始待ち時間が発生する．

以上より，各 $j \in \{1,\ldots,\lambda\}$ における機械 j にどのジョブをどのように割り当てたしたとしても，少なくとも 1 つ実行開始待ち時間が 3 以上になる機械が存在する．よって，H を充足する真理値割り当て f が存在しないとき，実行開始待ち時間が 2 以下となるスケジュールは存在しない． ■

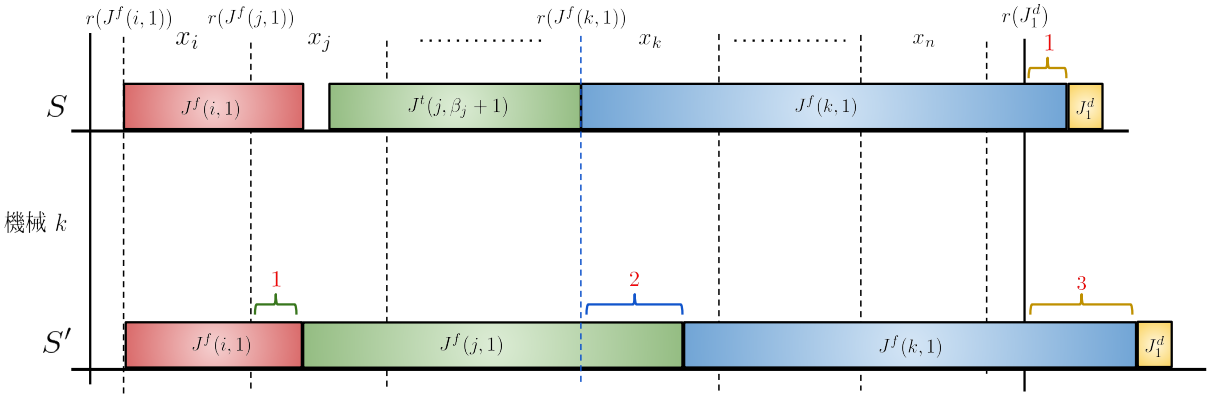


図 4.2: $h_k = (x_i, x_j, x_k)$ のとき，各リテラルに対応するジョブのスケジュール

スケジュール S $x_i = 0, x_j = 1, x_k = 0$ のとき，つまり，H を充足する真理値割り当てが存在する場合．

スケジュール S' $x_i = 0, x_j = 0, x_k = 0$ のとき，つまり，H を充足する真理値割り当てが存在しない場合．

補題 6 と補題 7 より，H を充足する真理値割り当て $f : X \to \{0, 1\}$ が存在することと，$I_{(X,H)}$ を入力とする最大実行開始待ち時間最小化問題に対して $\varphi(A, s) \leq 2$ を満たす実行可能なスケジュール (A, s) が存在することは，同値である．3 -SAT 問題の入力 (X, H) に基づく $I_{(X,H)}$ の生成に要する計算時間は，明らかに多項式時間であり，入力 $I_{(X,H)}$ の長さは，(X, H) の長さに関する多項式で表すことができる．したがって，無関連並列機械モデルにおいて機械数が入力の一部の場合， 最大実行開始待ち時間最小化問題は NP 困難である．

ここで，無関連並列機械モデルにおける最大実行開始待ち時間最小化問題のインスタンスと，スケジュールが与えられたとき，与えられたスケジュールにおける最大実行開始待ち時間が w 以下となるかの判定は，明らかに多項式時間で判定可能である．したがって，無関連並列機械モデルにおいて機械数が入力の一部の場合， 最大実行開始待ち時間最小化問題は NP 困難かつ，NP に属するので，NP 完全である．

4.2 計算複雑さのまとめ

本研究で，無関連並列機械モデルにおいて，機械数が入力の一部の場合，最大実行開始待ち時間最小化問題は，NP 完全であることが明らかになった．しかし，他の機械モデルにおける計算複雑さは明らかになっていない．

しかし，$w = 0$ のとき，この問題は JIT スケジューリング問題に対応し，単一機械モデルと同一機械モデルにおいて，多項式時間で判定できることが明らかになっている Cepek and Sung [2004] [1].

以下で，単一機械モデルにおける最大実行開始待ち時間最小化問題（決定問題）について，$w = 0$ のとき，以下の条件を満たすとき，決定問題の解は Yes，満たさないとき No であることは明らかである．

$$\forall J, J' \in \mathcal{J} \left[\big[r(J), r(J) + p(J)\big) \cap \big[r(J'), r(J') + p(J')\big) = \emptyset \right]$$

$w = 0$ となるスケジュールとは，すべてのジョブが処理開始可能時刻と 処理開始可能時刻 + 処理時間 の間で処理されなければならない．つまり，すべてのジョブに関して各ジョブの [処理開始可能時刻，処理開始可能時刻 + 処理時間] で表される範囲が被ってはいけない．この判定にかかる時間は高々 $\mathcal{O}(n^2)$ であることから，多項式時間で判定が可能である．同様に，同一並列機械モデルにおける最大実行開始待ち時間最小化問題（決定問題）についても多項式で判定可能である．

$|\mathcal{J}'| \leq |\mathcal{M}|$ を満たすとき，決定問題の解は Yes，満たさないとき No であることは明らかである．ただし $\mathcal{J}'$ は以下の通り．

$$\mathcal{J}' \subseteq \mathcal{J} \text{ s.t. } \forall J, J' \in \mathcal{J}' \big[[s(J), s(J) + p(J)) \cap [s(J'), s(J') + p(J')) \neq \emptyset \big]$$

第 2 章で紹介した機械モデルの関係より，各機械モデルにおけるスケジューリング問題は，互いに部分問題と拡張問題の関係にある．つまり，単一機械モデルにおける最大実行開始待ち時間最小化問題が NP 完全であれば，残りの機械モデルにおけるスケジューリング問題についても NP 完全であることが言える．しかし，本研究では，無関連並列機械モデルにおける NP 完全性のみ明らかにしたため，他の機械モデルにおける問題の計算複雑さは明らかでない．

単一機械モデルにおける最大実行開始待ち時間最小化問題（決定問題）は第 3 章で紹介した，処理開始可能時刻つき最大遅れ時間最小化問題と対応する．最大実行開始待ち時間最小化問題（決定問題）では，納期 = 処理開始可能時刻 + 処理時間 + w と定義した．したがって，処理開始可能時刻付き最大遅れ時間最小化問題により制限を加えた問題であると言える．つまり，最大実行開始待ち時間最小化問題は最大遅れ時間最小化問題の部分問題として捉えることができる．第 3 章より，処理開始可能時刻付き最大遅れ時間最小化問題が強 NP 困難であることから，最大実行開始待ち時間最小化問題も NP 困難であることが推測される．

第5章　解法の提案と実験的評価

第 5.1 節では，最大実行開始待ち時間最小化問題に対するヒューリスティックを紹介する．第 5.2 節では，最大実行開始待ち時間最小化問題に対する厳密解法を紹介する．第 5.3 節では，第 5.1 節で紹介したヒューリスティックの評価と分析を行う．また，第 5.2 節で紹介した厳密解法の計算時間の分析も行う．

5.1　ヒューリスティックの提案

本研究では，同一並列機械モデル，無関連並列機械モデルにおける最大実行開始待ち時間最小化問題に対して，貪欲アルゴリズムを用いた解法を提案する．

本研究の背景として，計算サーバーへのタスク割り当てを例に挙げて紹介した．計算サーバーへのタスク割り当ては最大実行開始待ち時間を最小化するオンラインスケジューリング問題として捉えることができる．オンラインスケジューリング問題とは，オンライン環境におけるスケジューリング問題のことである．オンライン環境とは，タスク（ジョブ）の情報があらかじめわからない環境をさす．つまり，現時点で，いくつのタスク（ジョブ）を処理する必要があるのか，各タスク（ジョブ）の処理時間はどのくらいなのかがわからない．よって，このような環境では，タスクをどの順番で処理するかではなく，タスクの処理要求が計算サーバーに届いた順で処理を開始するという方法が取られることは自然である．タスクの処理要求が計算サーバーに届いた順とは，各タスクの処理開始可能時刻順で処理することである．本研究の提案解法はこの考え方をもとにしている．

このような考え方は，貪欲的解法と呼ばれ，最適化問題に対する最も単純なアルゴリズムの 1 つとして知られており，Juraj Hromkovic [2004] [3] で紹介されている．バックトラッキングや局所探索と類似点は，実行可能解（$p_1 \ldots, p_n$）（$i \in \{1, \ldots, n\}[p_i \in P_i]$）の仕様が必要であり，任意の貪欲アルゴリズムは局所ステップの系列と見ることができる点である．しかし，貪欲アルゴリズムは 1 つの実行可能解からもう 1 つの実行可能解には遷移しない．まず，空の状態から始まり，仕様における 1 つの局所パラメータを永久に確定する．第 2 ステップでは，局所アルゴリズムは仕様における第 2 のパラメータを確定する．

この操作が実行可能解の完全な仕様に到達するまで繰り返される．貪欲アルゴリズムは，次の局所仕様を生成するために全ての可能性の中から最も有望と思われるパラメータを選択する．後で，どのような状況が起ころうと，この決定は決して変えられることはない．また，貪欲アルゴリズムはバックトラッキングで生成される木 $T_{\mathcal{M}(x)}$ における根から葉までのちょうど 1 つの路を実現している．空仕様は全ての実行可能解の集合 $\mathcal{M}(x)$ を考えており，$\mathcal{M}(x)$ が木 $T_{\mathcal{M}(x)}$ における根のラベルであることを意味している．第 1 のパラメータ p_1 を指定することは，$\mathcal{M}$ を集合 $S(p_1) = \{\alpha \in \mathcal{M}(x) \mid \alpha$ の仕様の第 1 パラメータは $p_1\}$ に制限することに対応する．この手続きを繰り返すと，実行可能解の集合の系列

$$\mathcal{M}(x) \supseteq S(p_1) \supseteq S(p_1, p_2) \supseteq \ldots, \supseteq S(p_1, p_2, \ldots, p_n)$$

を得る．ここで，$|S(p_1, p_2, \ldots, p_n)| = 1$ である．

以下の Heuristic は，同一並列機械モデルにおける最大実行開始待ち時間最小化問題に対する提案解法である．議論に先立ち表記を導入する．

- ジョブの集合 $\mathcal{J} = \{J_1, \ldots, J_n\}$.
- 処理開始可能時刻と処理開始可能時刻 + 処理時間 で表される区間が重なっているジョブの集合 $\mathcal{J}'$．つまり，$\mathcal{J}'$ は以下を満たす．

$$\mathcal{J}' \subseteq \mathcal{J} \text{ s.t. } \forall J, J' \in \mathcal{J}' \big[[s(J), s(J) + p(J)) \cap [s(J'), s(J') + p(J')) \neq \emptyset\big]$$

- 機械の集合 $\mathcal{M} = \{M_1, \ldots, M_m\}$.
- ジョブの処理開始可能時刻を返す関数 $r : \mathcal{J} \to \mathbb{N}$.
- ジョブの処理時間を返す関数 $p : \mathcal{J} \to \mathbb{N}$.

  ただし，無関連並列機械モデルのとき，$p : \mathcal{J} \times \mathcal{M} \to \mathbb{N}$ となる．
- スケジュールにおける最大実行開始待ち時間を返す関数 $w : S \to \mathbb{N}$.
- スケジュールにおけるジョブの完了時刻を返す関数 $C : \mathcal{J} \to \mathbb{N}$.
- スケジュール S はジョブの順列を表す.
- ．各 $M \in \mathcal{M}$ におけるスケジュール S_M の集合 $\mathcal{S}$．つまり，$\mathcal{S} = \{S_M \mid M \in \mathcal{M}\}$.

HEURISTIC

入力： $I = (\mathcal{J}, \mathcal{M}, r, p, w, C)$

出力： スケジュールの集合 $\mathcal{S}$.

Step 1. $|\mathcal{J}'| \leq |\mathcal{M}|$ を満たすとき，

Step 1.1. 各 $J \in \mathcal{J} \setminus \mathcal{J}'$ を任意の機械 $M \in \mathcal{M}$ に割り当てる．また，各 $J' \in \mathcal{J}'$ を $M \in \mathcal{M}$ に割り当てる．ただし，$\forall M, M' \in \mathcal{M}\big[S_M \cap S_{M'} = \emptyset\big]$ を満たす.

Step 2. $|\mathcal{J}'| \leq |\mathcal{M}|$ を満たさないとき，

Step 2.1. $\mathcal{J}$ を処理開始可能時刻の昇順でソートする.

Step 2.2. 各機械 $M \in \mathcal{M}$ のスケジュール $S_M \in \mathcal{S}$ におけるジョブの集合を $\mathcal{J}_{S_M}$ と表記する．ただし，

$$\forall M, M' \in \mathcal{M}\big[M \neq M' \Rightarrow \mathcal{J}_{S_M} \cap \mathcal{J}_{S_{M'}} = \emptyset\big]$$

$$\forall M, M' \in \mathcal{M}\big[M \neq M' \Rightarrow S_M \cap S'_M = \emptyset\big]$$

を満たす．つまり，$\mathcal{J}_{S_M}$ は $\mathcal{J}$ の分割，S_M は $\mathcal{S}$ の分割である.

Step 2.3. 各 $1 \leq i \leq n$ における J_i について以下の処理を繰り返す.

Step 2.3.1. 最小完了時刻を持つ機械集合の要素 $M_a \in \left\{\arg\min_{M \in \mathcal{M}} C(\mathcal{J}_{S_M})\right\}$ を 1 つ求める．ここで，$S_{M_a} := S_{M_a} \cup J_i$ とする.

Step 2.4. ここで，$\mathcal{S} = \{S_M \mid M \in \mathcal{M}\}$，$W = w(\mathcal{S})$ として，$\mathcal{S}, W$ を出力する.

上記 HEURISTIC は，すべてのジョブが JIT ジョブのとき，JIT スケジューリングに適用するアルゴリズムを適用する．それ以外のとき，処理開始可能時刻が最も早いジョブから順に，処理をしていない機械，または処理が一番早く終わった機械に割り当てる.

Step 1. では，各ジョブにおける処理開始可能時刻と処理開始可能時刻 + 処理時間 の区間が重なっているジョブの集合の要素数が機械数より少ない時，$\mathcal{J}'$ の各ジョブを別の機械に割り当てることで，実行開始待ち時間が 0 のスケジュールを作ることができる．それ以外のとき，処理開始可能時刻が最も早いジョブから順に，処理をしていない機械，または処理が一番早く終わった機械に割り当てる．Step 3. で，処理をしていない機械，または処理が一番早く終わった機械 を取得し M_a と定義している．その後，機械 M_a におけるスケジュール S_{M_a} にジョブ J_i を追加する.

この判定を追加したことで，インスタンスにおける最適解から得られる最大実行開始待ち時間が 0 の場合，上記 HEURISTIC でも，最大実行開始待ち時間が 0 となるスケジュールを作ることができる．

5.2 厳密解法の提案

同一並列機械モデル，無関連並列機械モデルにおける最大実行開始待ち時間最小化問題に対する厳密解法を開発した．厳密解法の開発においてよく用いられる発想の 1 つに，分枝限定法を用いる，という発想がある．

分枝限定法は最適化問題に対するアルゴリズムを設計するための方法である．この方法は計算時間がどうであろうと，無条件で最適解を見つけたいときに用いる．分枝限定法は，すべての実行可能解の空間をしらみつぶしに探索する手法のバックトラッキングに基づいている．分枝限定法の大まかなアイデアは，しらみつぶし探索において，解を生成するある部分に達したとき，その部分に最適解が含まれていないことがわかったときには，実行可能解の空間のその部分の探索を省略することによってバックトラッキングを早くすることである．

ここで，バックトラッキングを適用するための表記法を導入する．$\mathcal{M}(x)$ を最適化問題の入力インスタンス x に対するすべての実行可能解の集合とする．$T_{\mathcal{M}(x)}$ を以下のような性質を持つラベル付き木と定義する．

- $T_{\mathcal{M}(x)}$ の任意の頂点 v は集合 $S_v \subseteq \mathcal{M}(x)$ によってラベル付けられている．
- $T_{\mathcal{M}(x)}$ の根は $\mathcal{M}(x)$ によってラベル付けられている．
- $T_{\mathcal{M}(x)}$ において $v_1, \ldots, v_m$ が v の親の全ての子であるならば，

  $$\forall i, j \in \{1, \ldots, m\} \left[i \neq j \Rightarrow S_v = \bigcup_{i=1}^{m} S_{v_i} \wedge S_{v_i} \cap S_{v_j} = \emptyset \right]$$

  が成り立つ．つまり，v は S_v の分割である．
- $T_{\mathcal{M}(x)}$ の各葉 u に対して，$|S_u| \leq 1$．つまり，葉が $\mathcal{M}(x)$ の実行可能解に対応している．

バックトラッキングは，ラベルの付いた根付き木 $T_{\mathcal{M}(x)}$ に対する深さ優先探索，あるいは幅優先探索とみなすことができる，ここで葉は $\mathcal{M}(x)$ からの実行可能解でラベル付けされており，$T_{\mathcal{M}(x)}$ の全ての内部頂点 v には，v を根とする部分木 T_v の葉のラベルが

付いた全ての実行可能解を含む $S_v \subseteq \mathcal{M}(x)$ でラベル付けされている．分枝限定法は，アルゴリズムが v を訪れた時点で，T_v が最適解を持たないこと決定できるときに，$T_{\mathcal{M}(x)}$ から T_v を切り取ることに他ならない．このアプローチの効率はアルゴリズムの実行中に切断され得る $T_{\mathcal{M}(x)}$ の部分木の量とサイズに依存する．

分枝限定法は多くの NP 困難な組合わせ最適化問題に対して，その最適解を求めるための最も良い枠組みとして知られており，Land and Doig [1960] [6] によって提案され，Little, Murty, Sweeney and Karel [1963] [8] によって TSP にはじめて適用されている．

分枝限定法の最も単純な版は，頂点 v に到達したとき，それまでに見つかった最良のコストと T_v の S_v における実行可能解の最小または最大（ここでは最大）のコストを比較する．それまでの最良のコストが評価された範囲のどのコストよりも明らかに良ければ，T_v を切り取る（すなわち，T_v の探索を中断する）．

この節では，同一並列機械モデルにおける最大実行開始待ち時間最小化問題に対する厳密解法に用いた分割生成アルゴリズムと分枝限定法アルゴリズムを紹介する．分枝限定法によるアプローチの効率はアルゴリズムの実行中に切断される $T_{\mathcal{M}(x)}$ の部分木の量とサイズ，プログラムに依存する．

以下では，分割の生成方法，分枝限定法のアルゴリズムと工夫部分について，紹介する．その後，アルゴリズムの紹介を行う．分割の生成方法については Kreher. Donald L. [1998] [4] によって，紹介されている．

同一並列機械の場合，全ての機械の性能が同じであることから，ジョブをどの機械に割り当てたとしても，処理時間は変わらない．そのため，同一並列機械では，どの機械に割り当てるかではなく，どのジョブと同じ機械で処理するかを考える必要がある．各ジョブの処理は任意の 1 機械で処理を完了するため，同一並列機械へのジョブの割り当ては，ジョブの分割として捉えることができる．分割の定義は以下の通り．$\mathcal{J}$ の分割 Π の要素 $\pi \in \Pi$ に対して，以下の条件を満たす Π.

- $\bigcup_{\pi \in \Pi} \pi = \mathcal{J}$
- $\forall \pi, \pi' \in \Pi \big[\pi \neq \pi' \Rightarrow \pi \cap \pi' = \emptyset \big]$

議論に先立ち，表記を導入する．コストの下限を返す関数を $cost : \mathcal{J} \to \mathbb{N}$ とする．また，$M = \pi$ は分割の要素 π に機械 M が対応することを意味する．

分割生成アルゴリズムの改良

1. $|\Pi| = |\mathcal{M}|$ を満たす分割 Π のみ生成した．

   機械数が m のとき，生成した分割の要素数が m 未満，または m より多いとき，その分割に対する探索を行う必要がないため，そのような分割を生成しないアルゴリズムに改良することで，考慮する分割を減らすことを可能にした．

2. 各 $\pi \in \Pi$ における $\mathcal{J}_\pi$ のコストの下限 $cost(\mathcal{J}_\pi)$ を評価し，分割 Π において，スケジュールを生成するかどうかの判定を追加した．

3. 各 $\pi \in \Pi$ における $cost(\mathcal{J}_\pi)$ を降順でソートし，順に π に対応する機械にジョブ $J \in \mathcal{J}_\pi$ を割り当てる．

   各 $\pi \in \Pi$ における $cost(\mathcal{J}_\pi)$ の値が大きい機械から順に割り当てることで，その分割において，目的関数に影響を与える可能性が高いスケジュールから順に考慮することができるため，探索の中断を早い段階で行うことができる．

分枝限定法アルゴリズムの改良

1. $cost(\mathcal{J}_{M_a} \setminus \mathcal{J}_{S_{M_a}})$ を評価し，探索を続行するか中断するかの判定を追加した．機械 $M_a = \pi \in \Pi$ において，M_a に割り当てられたジョブの集合を $\mathcal{J}_{S_{M_a}}$ とする．

   割り当てられたジョブだけでなく，割り当てられていないジョブに対するコストの下限の評価を加えることで，探索の中断，続行の判定を増やした．

以下では，上記改良を加えた分割生成アルゴリズムと，分枝限定法アルゴリズムの紹介を行う．議論に先立ち表記の導入を行う．

- ジョブの集合 $\mathcal{J} = \{J_1, \ldots, J_n\}$
- 機械の集合 $\mathcal{M} = \{M_1, \ldots, M_m\}$
- ジョブの処理開始時刻を返す関数 $s : \mathcal{J} \to \mathbb{N}$
- ジョブの処理開始可能時刻を返す関数 $r : \mathcal{J} \to \mathbb{N}$
- ジョブの処理時間を返す関数 $p : \mathcal{J} \to \mathbb{N}$
- スケジュールにおける最大実行開始待ち時間を返す関数 $w : S \to \mathbb{N}$
- スケジュールにおけるジョブの完了時刻を返す関数 $C : \mathcal{J} \to \mathbb{N}$
- 各 $j \in \{1, \ldots, m\}$ に対する機械 $M_j \in \mathcal{M}$ におけるスケジュール S_{M_j} の集合 $\mathcal{S}$

- それまでのスケジュールにおける最良の解 W
- 要素数 n の分割を表す配列 b
- 要素数 n の配列 $b_{\max}$ は 各 $1 \leq i \leq n$ に対して，i 番目の要素は，それまでのジョブが割り当てられた機械の最大数を格納している．つまり，$b_{\max}[i] = \max_{1 \leq j \leq i-1} b[j]$.

StrictSolutionMethod

入力： $I = (\mathcal{J}, \mathcal{M}, s, r, p, w, C, W)$

出力： スケジュールの集合 $\mathcal{S}$

Step 1. I を入力とし，Heuristic を実行する．出力を $\mathcal{S}$ とする．このとき，$W = \max_{S \in \mathcal{S}} w(S)$ とする．$W = 0$ のとき，スケジュール $\mathcal{S}$ を出力して処理を終了する．

Step 2. 組 $(b, b_{\max}, n, m)$ を入力として，increment を実行し，出力が false となるまで，以下の処理を繰り返す．

Step 2.1. (b) を入力として，sort を実行する．

Step 2.2. 各 $1 \leq i \leq n$ に対して以下の処理を繰り返す．

Step 2.2.1 ジョブの集合 $\mathcal{J}'$ を定義する．ただし $\mathcal{J}' = \emptyset$.

Step 2.2.2. 各 $1 \leq j \leq m$ に対して，以下の処理を繰り返す．

Step 2.2.2.1. $order[i] = j$ のとき，$\mathcal{J}' := \mathcal{J}' \cup J_i$ とする．

Step 2.2.3. 組 $(\mathcal{J}', s, r, p, w, C, 0)$ を入力として，BranchAndBound を実行する．

Step 2.2.4. BranchAndBound の出力を S とする．このとき，$\mathcal{S} := \mathcal{S} \cup S$ とする．

Step 3. スケジュール $\mathcal{S}$ を出力する．

上記 StrictSolutionMethod では，normalize と increment により，分割 Π の要素数 が 機械数 m と等しくなるように，分割を生成している．その後 BranchAndBound により，各 $\pi \in \Pi$ に対して，順列の生成を行っている．

Algorithm： normalize

入力： $I = (b, b_{\max}, m)$

出力：分割 b

Step 1. 各 $1 \leq i \leq n$ に対して以下の処理を繰り返す.

Step 1.1. $b_{\max}[n-i] \geq m-i$ のとき，処理を終了する.

Step 1.2. $b[n-i] := m-i$，$b_{\max}[n-i] := m-i$

上記 NORMALIZE は，INCREMENT で生成した b のすべての要素が m 未満のとき，割り当てていない機械が存在するので，その分割 b を m となる要素を少なくとも 1 つ持つ b に変換している．Step 1.1. では，$n-i$ 番目の 1 つ前のジョブまでが割り当てられた機械番号の最大値が m 以上のとき，すでに要素 m を持っているので，変換せずに出力している.

Step 1.2. 割り当てられていない機械が存在するので，各 $0 \leq i \leq n-1$ に対して，$n-i$ から順に $b[n-i] := m-i$ とすることで，分割の要素が m となるように調整している.

ALGORITHM： INCREMENT

入力：$I = (b, b_{\max}, n, m)$

出力：b が更新されたとき TRUE を出力し，それ以外のとき FALSE を出力する.

Step 1. 各 $n \geq i \geq 1$ に対して以下の処理を繰り返す.

Step 1.1 $b[i] = \min\{b_{\max}[i]+1, m\}$ のとき，$b[i] := 1$ とする.

Step 1.2 $b[i] \neq \min\{b_{\max}[i], m\}$ のとき，$b[i] := b[i]+1$ とする.

Step 1.2.1 $b[i] > b_{\max}[i]$ のとき，各 $i+1 \leq j \leq n$ に対して，$b_{\max}[j] := b[i]$ とする.

Step 1.2.2 $I = (b, b_{\max}, m)$ を入力として，NORMALIZE を実行する.

Step 1.2.3 TRUE を出力する.

Step 2. FALSE を出力する.

上記 INCREMENT は，辞書式順で分割を生成している．配列 b は分割を表しており，i 番目の要素 $b[i]$ は，ジョブ J_i が機械 $b[i]$ に割り当てられていることを表す．ここで，分割の要素数は $\max_{1 \leq i \leq n} b[i]$ で表すことができる.

Step 1.1. では，各 $n \geq i \geq 1$ における $b[i]$ が $\min\{b_{\max}[i]+1, m\}$ のとき，それ以上新たな機械を追加することができないので，リセットしてる.

Step 1.2. では，i 番目のジョブを割り当てる機械を新たに追加することができるので，$b[i] := b[i] + 1$ としている．そのとき，Step 1.2.1 で i 番目までの機械の最大数を更新している．

新たな分割を生成したときに，TRUE を，生成できなかったときに FALSE を返している．

ALGORITHM： SORT

入力： $I = (b)$

出力： 機械への割り当て順を表す配列 $order$

Step 1. 要素数 m の配列 $cost$ と $order$ を定義する．ただし，$1 \leq j \leq m$ に対して $order[j] = j$ とする．

Step 2. 各 $1 \leq i \leq n$ に対して以下の処理を繰り返す．

Step 2.1. ジョブの集合 $\mathcal{J}'$ を定義する．ただし $\mathcal{J}' = \emptyset$.

Step 2.2. $1 \leq j \leq m$ に対して以下の処理を繰り返す．

Step 2.2.1. $b[i] = j$ のとき，$\mathcal{J}' := \mathcal{J}' \cup J_i$ とする．

Step 2.3. $I = (\mathcal{J}', S_{M_j})$ を入力として，EVALUATION を実行する．

Step 2.4. EVALUATION により出力されたスケジュールを S'_{M_j} とし，$cost[j] := w(S'_{M_j})$ とする．

Step 3. $cost$ の降順となるように，$order$ をソートする．

Step 4. $order$ を出力する．

ALGORITHM EVALUATION

入力： $I = (\mathcal{J}', S)$

出力： スケジュール S

Step 1. 各 $i \in \{1, \ldots, |\mathcal{J}'|\}$ における J'_i について，$p(J'_i) = \min_{j \in \{1,\ldots,|\mathcal{J}'|\}} p(J'_j)$ とする．

Step 2. 各 $1 \leq i \leq |\mathcal{J}'|$ に対して以下の処理を繰り返す．

Step 2.1. $S := S \cup J'_i$ とする．

Step 3. S を出力する．

上記 EVALUATION は，機械に割り当てられていないジョブの集合 $\mathcal{J}'$ について，部分問題に対する多項式アルゴリズムを用いて，解を求めている．

それまでのスケジュールにおける最良の解のコスト W と，スケジュールされていない残りのジョブを最適にスケジュールして得られた解 w' を比較して $W \leq w'$ であれば，探索を中断することができ，効率を上げることができる．ここで，最適なスケジュールの求める方法として，第 3 章で紹介した部分問題に対する多項式アルゴリズムを用いる．また，残りのジョブを本来の処理時間で最適にスケジュールして得られる解を W_{opt} とする．

すべてのジョブに関して，処理時間が一定のとき，処理開始可能時刻の昇順で処理することで最適なスケジュールが求まる．このとき，EVALUATION では全てのジョブの処理時間を残りのジョブの最小の処理時間 $p_{\min}$ とした．ここで，残りのすべてのジョブの中で最大の処理時間を $p_{\max}$ としたとき，すべてのジョブの処理時間 p は $p_{\min} \leq p \leq p_{\max}$ である．ここで,，設定する処理時間が $p_{\min}$ より大きいとき，設定した処理時間より小さい値を持つジョブが少なくとも 1 つ存在する．このとき，$w' < W_{opt}$ とならない可能性がある．$W_{opt} < w'$ のとき，最低でも w' 以上のコストがかかることを表しているので，最適なスケジュールを含む部分木を切り取ってしまう可能性がある．そのため，残りのジョブの処理時間を $p_{\min}$ として設定することで，最適なスケジュールを含む部分木を切り取ることなく探索できる．

ALGORITHM： BRANCHANDBOUND

入力： $I = (\mathcal{J}', r, p, w, C, W, i)$

出力： スケジュール S

$i = |\mathcal{J}'|$ のとき，

Step 1. $W = w(S)$ とし，スケジュール S を出力する．

$i \neq |\mathcal{J}'|$ のとき，

Step 2. $\mathcal{J}'$ を処理開始可能時刻の昇順でソートする．

Step 3. 各 $1 \leq j \leq |\mathcal{J}'|$ に対して以下の処理を繰り返す．ただし，$\mathcal{J}'$ における j 番目のジョブを J'_j と表記する．

Step 3.1 $S := S \cup J'_j$ とし，$\mathcal{J}'$ から J'_j を取り除く．

Step 3.2 $\mathcal{J}'$ に対して，$(\mathcal{J}', S)$ を入力として，EVALUATION を適用し，出力されたスケジュールを S' とする．

Step 3.2.1 $W \leq w(S')$ のとき，$(\mathcal{J}', r, p, w, C, W, i+1)$ を入力として BRANCHANDBOUND を実行する．

Step 3.3 J'_j を $\mathcal{J}'$ の取り除いた場所に戻す．

上記 BranchAndBound は，再帰的に順列の生成を行っている．Step 3.2. では，機械に割り当てられていないジョブの集合 $\mathcal{J}'$ に対して，部分問題に対する多項式アルゴリズムを用いて出力したスケジュール S' における最大実行開始待ち時間 $w(S')$ がそれまでの最良の解 W より良いとき，探索を続行している．全てのジョブをスケジュールしたとき，Step 1. で解の更新を行っている．それ以降の探索は更新した解を用いて判定を行う．

5.3 提案解法の評価と分析

同一並列機械モデル，無関連並列機械モデルにおける最大実行開始待ち時間最小化問題に対して，本研究で提案したヒューリスティックの実験的評価を行った．評価方法は，各インスタンスに対して，本研究で提案するアルゴリズムにより出力された解から得られた最大実行開始待ち時間 W_h と，厳密解法により出力された最適解から得られた最大実行開始待ち時間 W_{opt} を比較するものである．また，W_h/W_{opt} を competitive ratio と表記する．本研究では，ヒューリスティックと厳密解法に対して以下を評価の軸として実験を行った．

ヒューリスティックの評価

軸 1： 各インスタンスにおけるジョブの処理時間の幅と competitive ratio の関係

軸 2： 各インスタンスにおけるジョブの処理時間の幅と計算時間の関係

軸 3： 同一並列機械モデルと無関連並列機械モデルの competitive ratio の比較

軸 4： 同一並列機械モデルと無関連並列機械モデルの計算時間の比較

厳密解法の評価

軸 5： 分割生成アルゴリズムの改良前と，改良後の計算時間の比較

軸 6： 分枝限定法アルゴリズムの改良前と，改良後の計算時間の比較

実験結果の紹介に先立ち，得られた competitive ratio の値がどの程度であれば，アルゴリズムの質が良いと言えるかという点について紹介する．

表 5.1 と同一並列機械の集合 $\mathcal{M}(|\mathcal{M}| = 2)$ がインスタンスとして与えられたとき，最適なスケジュールとそうでないスケジュールは以下の通り．

$\mathcal{J}$	$r(J)$	$p(J)$
J_1	2	8
J_2	3	7
J_3	11	11
J_4	16	10
J_5	23	10
J_6	26	11

表 5.1: 各ジョブにおける処理開始可能時刻と処理時間

最適なスケジュール $S_1 : (J_1, J_3, J_5)$，$S_2 : (J_2, J_4, J_6)$．このとき，$w(\mathcal{S}) = 0$ となる．ただし，$\mathcal{S} = S_1 \cup S_2$．

最適でないスケジュール $S_1 : (J_1, J_3, J_6)$，$S_2 : (J_2, J_4, J_5)$．このとき，$w(\mathcal{S}) = 3$ となる．ただし，$\mathcal{S} = S_1 \cup S_2$．

上記のとき，COMPETITIVE RATIO = INFINITY となり，最適でないスケジュールを生成したアルゴリズムの質は無限大となる．以下の実験結果は，COMPETITIVE RATIO = INFINITY となりうる状況下で得られたものである．つまり，COMPETITIVE RATIO = 定数 のとき，アルゴリズムの質は悪くないと言える．

第6章　結論

6.1　研究成果

最大実行開始待ち時間最小化問題とは，各ジョブの処理開始可能時刻を制約とし，処理開始時刻と処理開始可能時刻の最大値の最小化を目的としたスケジューリング問題である．以下が本研究の研究成果である．

計算複雑さ　3-SAT からの還元により，無関連並列機械モデルにおいて，機械数が入力の一部の場合，最大実行開始待ち時間最小化問題の NP 完全性を示した．

最大実行開始待ち時間最小化問題の部分問題である，JIT ジョブ荷重和最大化問題における 3-SAT からの還元手法に基づき，最大実行開始待ち時間最小化問題の NP 完全性を示した．

解法　同一並列機械モデル，無関連並列機械モデルにおける最大実行開始待ち時間最小化問題に対して，ヒューリスティックと厳密解法を開発し，有効性を示した．

貪欲アルゴリズムに基づいたヒューリスティックを開発した．ヒューリスティックにより出力された解から得られた最大実行開始待ち時間を W_h，厳密解法より出力された最適解から得られた最大実行開始待ち時間を W_{opt} としたとき，$\max\left\{W_h/W_{opt}\right\} =$??? の結果が得られた．

厳密解法における計算効率の向上　厳密解法に用いた分割生成アルゴリズムと分枝限定法アルゴリズムを改良し，計算効率を向上させた．

分割の要素数 = 機械数となる分割のみを生成するアルゴリズムに改良した．この改良により，考慮する分割の数を減らし，計算効率を向上させた．また，分枝限定法アルゴリズムに処理開始可能時刻付き最大遅れ時間最小化問題の部分問題に対する多項式アルゴリズムの概念を導入した．この改良により，列挙する順列を減らし，計算効率を向上させた．その結果，同じインスタンスに対して，計算時間を約 ??? 倍にすることに成功した．

まとめ　本研究では，無関連並列機械モデルにおいて，機械数が入力の一部の場合，最大実行開始待ち時間最小化問題が NP 完全であることを証明した．しかし，単一機械モデル，同一並列機械モデル，一様並列機械モデルにおける計算複雑さは明らかでない．本研究で示した還元方法が，その他の機械モデルにおける計算複雑さの証明の足がかりになるものと期待する．

6.2　今後の課題

計算複雑さ　本研究で，無関連並列機械モデルにおいて，機械数が入力の一部の場合，最大実行開始待ち時間最小化問題は NP 完全であることを証明した．しかし，単一機械モデル，同一並列機械モデル，一様並列機械モデルにおける計算複雑さは明らかでない．これらの機械モデルにおける計算複雑さを明らかにし，問題の難しさに与える影響の特徴づけは，今後の課題である．

解法　本研究で開発した解法は，単一機械モデル，同一並列機械モデル，無関連並列機械モデルにおける最大実行開始待ち時間最小化問題に対して適用できるものである．一様並列機械モデル，における解法の開発は今後の課題である．

また，Web サービスを運用している会社の計算サーバーへのタスク割り当てを例として挙げたが，　現実では，計算サーバーに割り当てられるタスクは，Web サービス内での処理に限られるため，タスクの処理時間は大きくは変わらないと考えられる．さらに，タスクの処理時間の種類もある程度把握できると考えられる．本研究では，タスクの処理時間の分類についての分析は行っていない．より，現実的に計算サーバーへのタスク割り当てを考える場合，タスクの処理時間の分類についての分析を行い，最大実行開始開始待ち時間の部分問題に対する効率的なアルゴリズムの開発を行うことが考えられる．

厳密解法における計算効率の向上　本研究では，全ての機械モデルにおける最大実行開始待ち時間最小化問題は NP 困難であるという推察のもと，厳密解法に用いた分割生成アルゴリズムと分枝限定法アルゴリズムに改良を加えた．

無関連並列機械モデル以外のモデルにおける多項式アルゴリズム，もしくは擬似多項式アルゴリズムが存在すれば，分析規模を大幅に拡張することができる．この課題は，計算複雑さの解明により，計算時間の短縮に繋がると考えられる．

第7章　付録

7.1　実験結果

参考文献

[1] Cepek, O., Sung, S.C.: Just-in-time scheduling with periodic time slots. Scientiae Mathematicae Japonicae Online 10, 431–437 (2004)

[2] Garey, M.R., Johnson, D.S.: Computers and Intractability; A Guide to the Theory of NP-Completeness. W. H. Freeman And Co, New York, NY, USA (1990)

[3] Hromkovic, J.: Algorithmics for Hard Problems Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics. Springer-Verlag Berlin Heidelberg (2004)

[4] L., K.D.: Combinatorial algorithms : generation, enumeration, and search. CRC Press Boca Raton London New York Washington, D.C. (1998)

[5] Lageweg, B., Lenstra, J., Kan, A.: Minimizing maximum lateness on one machine: Computational experience and some applications (1976)

[6] Land, A.H., Doig, A.G.: An automatic method of solving discrete programming problems. The Econometric Society 28, 497–520 (1960)

[7] Lawler, E.L.: Optimal sequencing of a single machine subject to precedence constraints. Journal Management Science 19, 544–546 (1973)

[8] Little, Murty, S., Karel: An algorithm for the traveling salesman problem. Operations Research 11, 972–989 (1963)

[9] Sung, S.C., Vlach: Maximizing weighted number of just-in-time jobs on unrelated parallel machines. Jurnal of Scheduling 8, 453–460 (2005)

謝辞

本研究を進めるにあたり指導教員の宋教授には，研究に対する助言や熱心な指導をしていただきましたことを心から感謝致します．またゼミや日常で多くの知識や示唆をいただいた高橋先生，吉山先輩，田中先輩，丹羽先輩，牧石先輩と研究室の同期の方々に深く感謝致します

2017 年 1 月 31 日 天本　祐希