離散最適化基礎論

ジョブ・スケジューリングのアルゴリズム

第 6 回

リスト・スケジューリング

岡本 吉央　(電気通信大学)

okamotoy@uec.ac.jp

2024 年 11 月 19 日

最終更新：2024 年 11 月 20 日　17:27

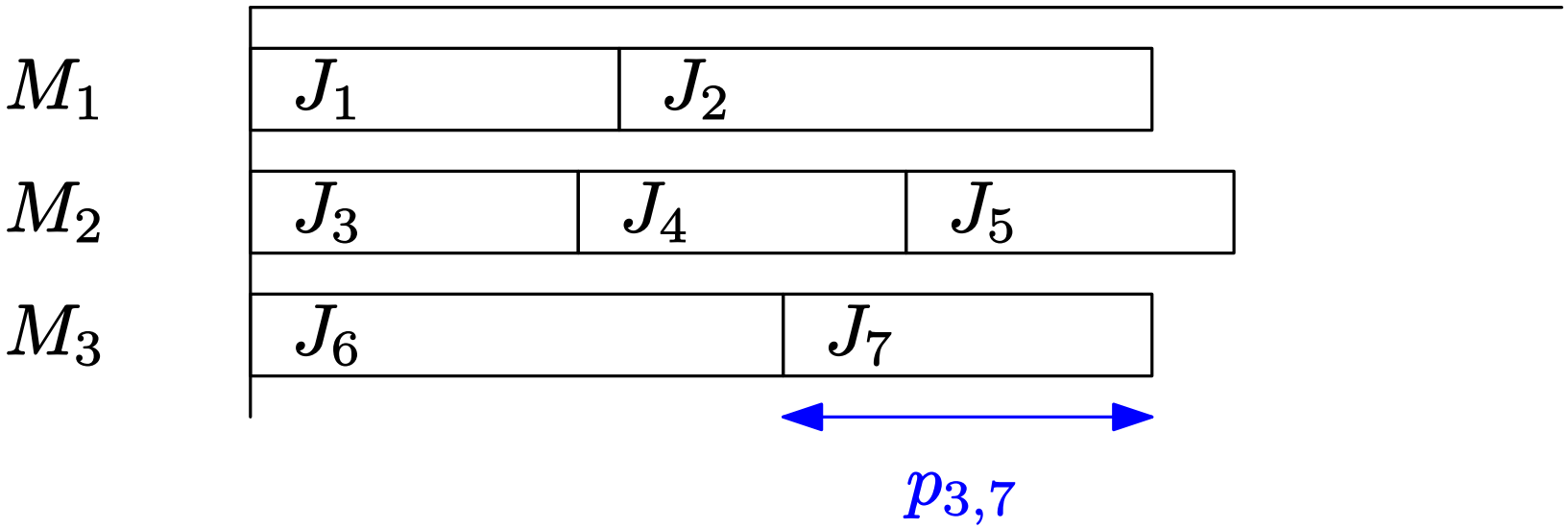
スケジュール (前半)

1. スケジューリング問題の分類 (10/1)
 * 休み (出張) (10/8)
 * 休み (体育祭) (10/15)
2. 整列による解法 (10/22)
3. 動的計画法 (10/29)
4. NP 困難性と計算量の分類 (11/5)
5. 計算複雑性による問題の分類 (11/12)
6. **リスト・スケジューリング** (11/19)

スケジュール (後半)

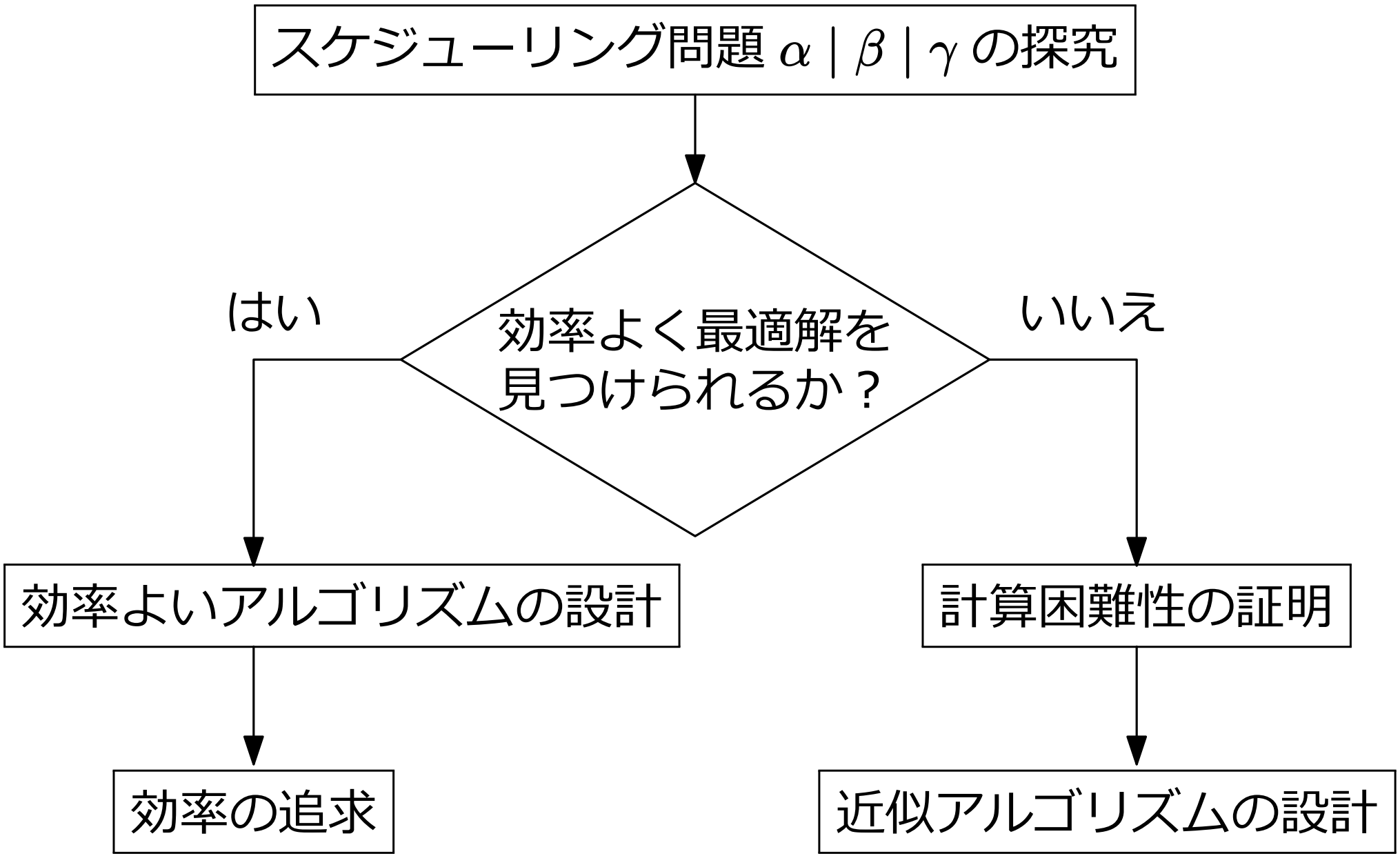
7. 先行制約：基礎 (11/26)
* 休み (秋ターム試験) (12/3)
8. 先行制約：多機械 (12/10)
9. 先行制約：他の半順序 (12/17)
10. ショップ・スケジューリング：基礎 (12/24)
* 休み (冬季休業) (12/31)
11. ショップ・スケジューリング：機械数が定数 (1/7)
12. ショップ・スケジューリング：機械数が可変 (1/14)
13. 近似可能性と近似不可能性 (1/21)
14. 多項式時間近似スキーム (1/28)
* なし (2/4)

ジョブ・スケジューリングの構成要素

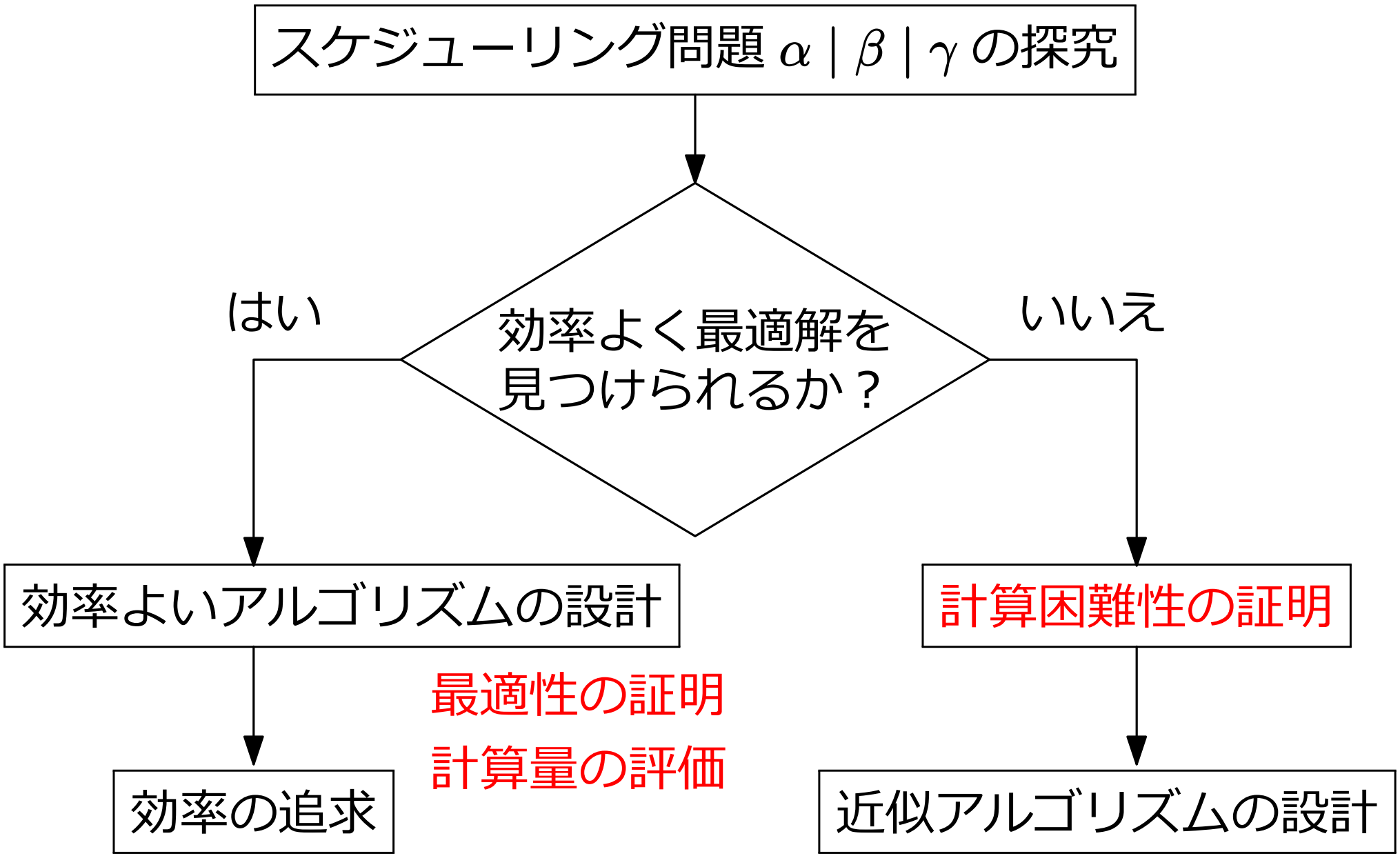


- 機械の集合　$\mathcal{M} = \{M_1, M_2, \ldots, M_m\}$
  添字 $i \in \{1, 2, \ldots, m\} =: [m]$
- ジョブの集合 (仕事)　$\mathcal{J} = \{J_1, J_2, \ldots, J_n\}$
  添字 $j \in \{1, 2, \ldots, n\} =: [n]$
- 処理時間　$p \in \mathbb{Q}_{\geq 0}^{m \times n}$ (非負有理数の行列)
  $p_{ij} = M_i$ における J_j の処理時間

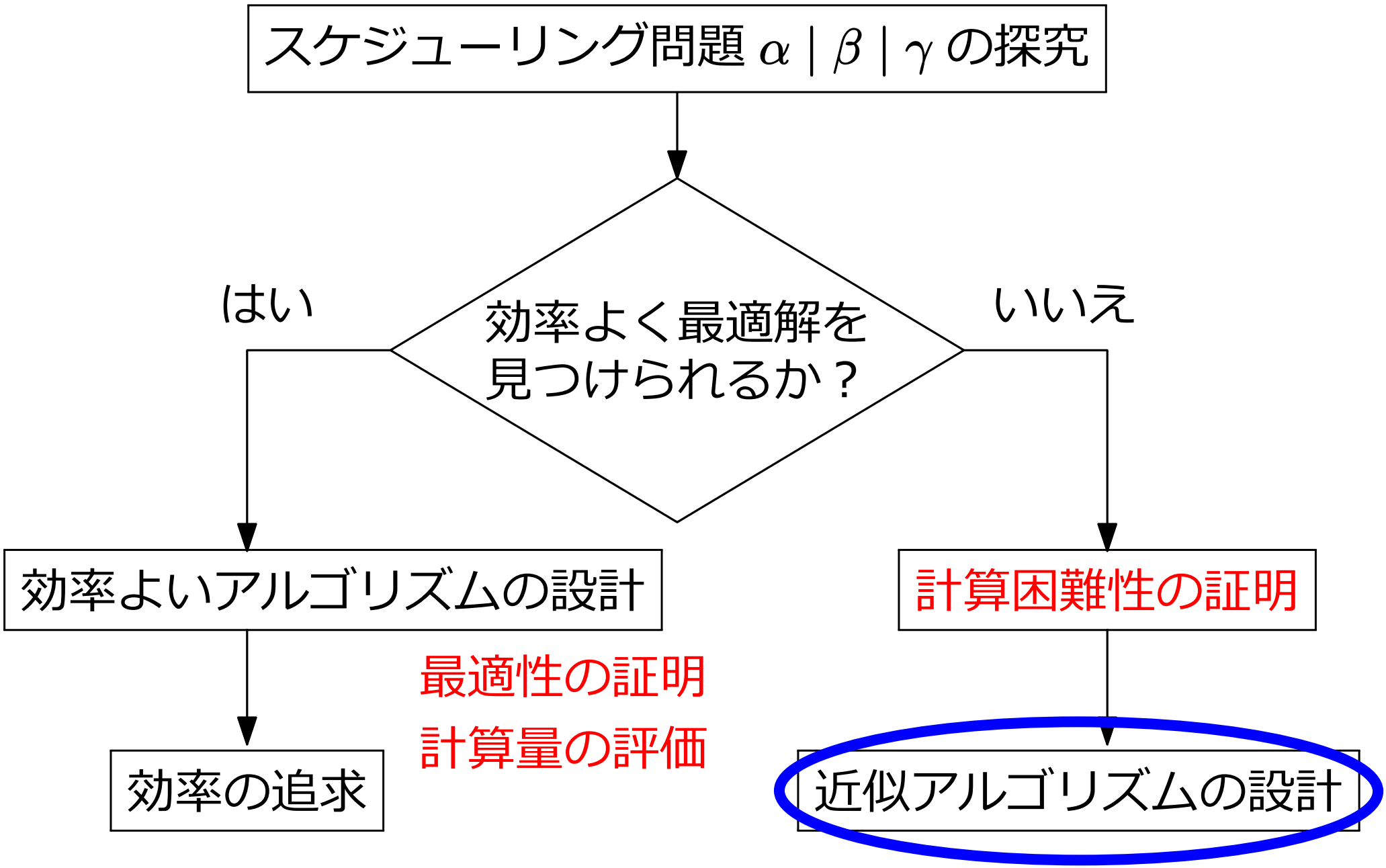
この講義の目標：第1回より



この講義の目標：第1回より



この講義の目標：第1回より



今日の内容

目標

リスト・スケジューリング を例にして
近似アルゴリズムの解析手法 を身に付ける

- 重要概念：近似保証，近似比
- 対象とする問題：$P \mid\mid C_{\max}$，$P \mid\mid \sum C_j$，$P \mid\mid \sum w_j C_j$

本日の内容

1. **$P \mid\mid C_{\max}$ に対するリスト・スケジューリング**
2. $P \mid\mid \sum C_j$ に対するリスト・スケジューリング
3. $P \mid\mid \sum w_j C_j$ に対するリスト・スケジューリング

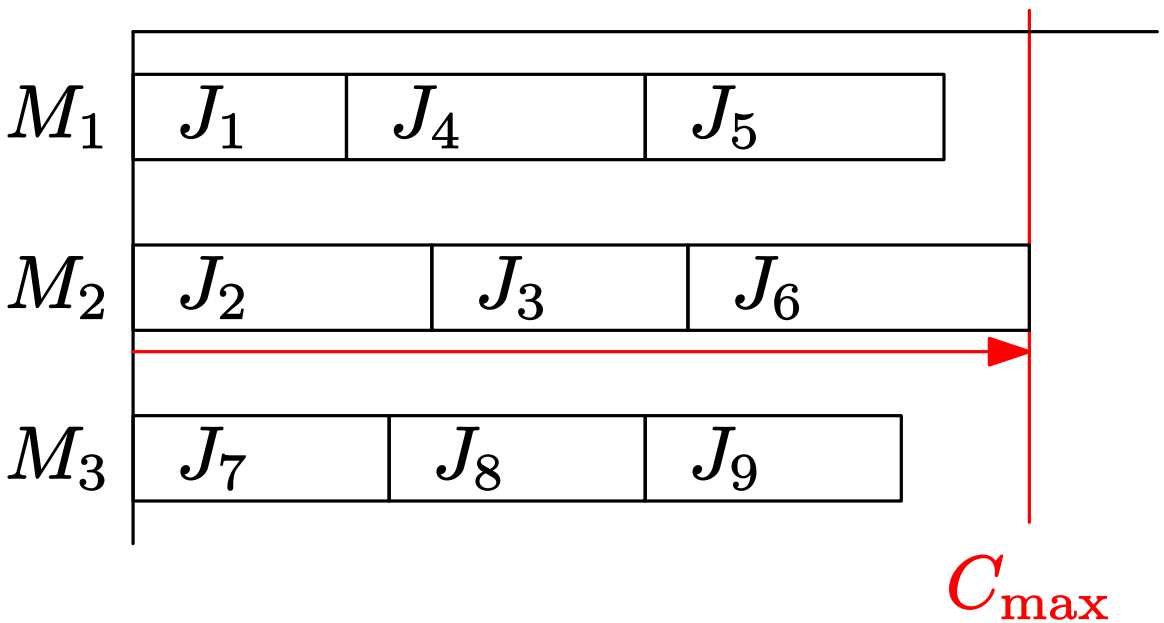
- R. L. Graham, Bounds for certain multiprocessing anomalies. *Bell System Technical Journal* 45 (1966) pp. 1563–1581.
- R. L. Graham, Bounds on multiprocessing timing anomalies. *SIAM Journal of Applied Mathematics* 17 (1969) pp. 263–269.

機械の環境

- 一様並列機械，機械数 $= m$ (入力の一部)
  - $p_j =$ ジョブ J_j の処理時間

最適化する目的

- 最大完了時刻の最小化

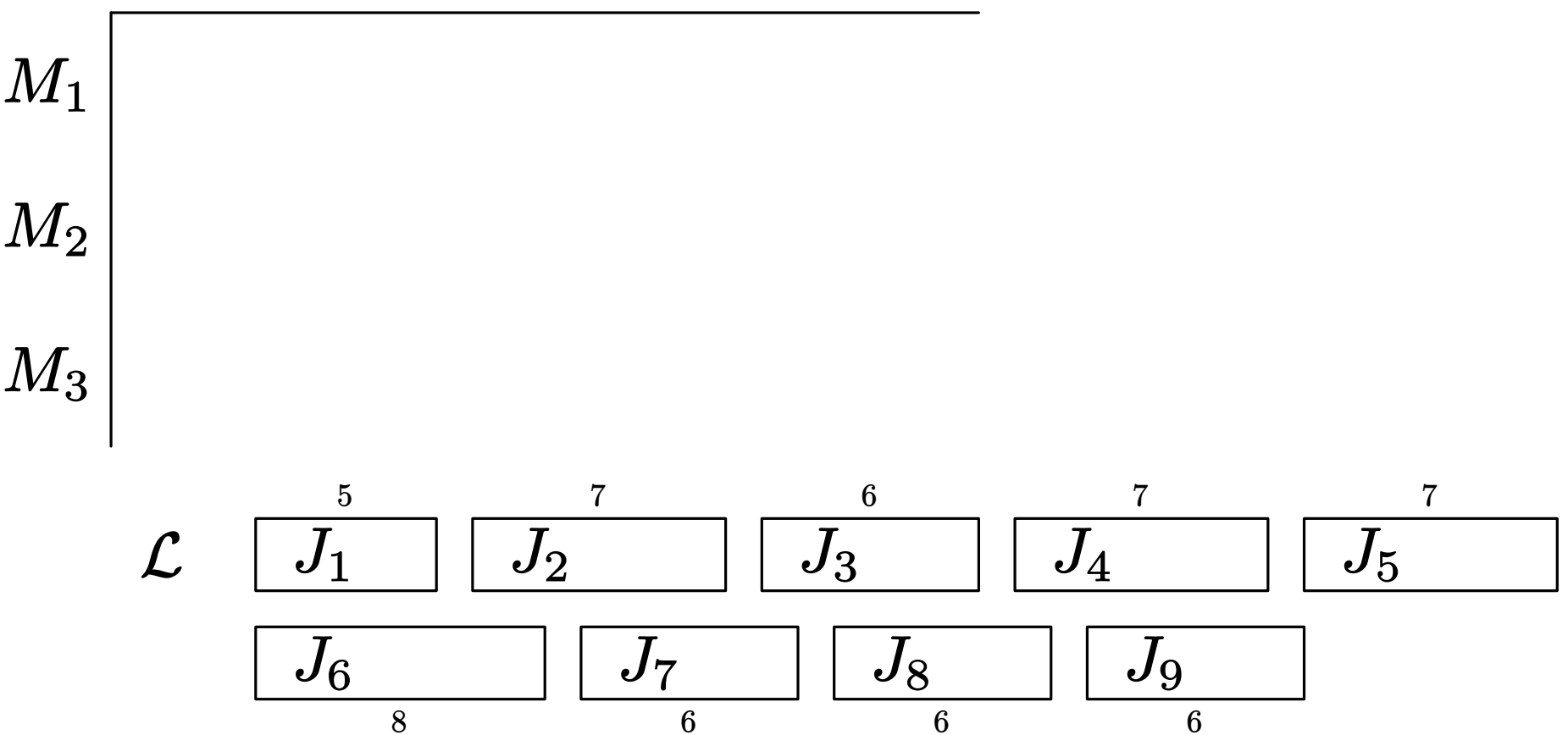


復習：問題 $P \,||\, C_{\max}$ は強 NP 困難

リスト・スケジューリング：考え方

リスト・スケジューリング：ひな型

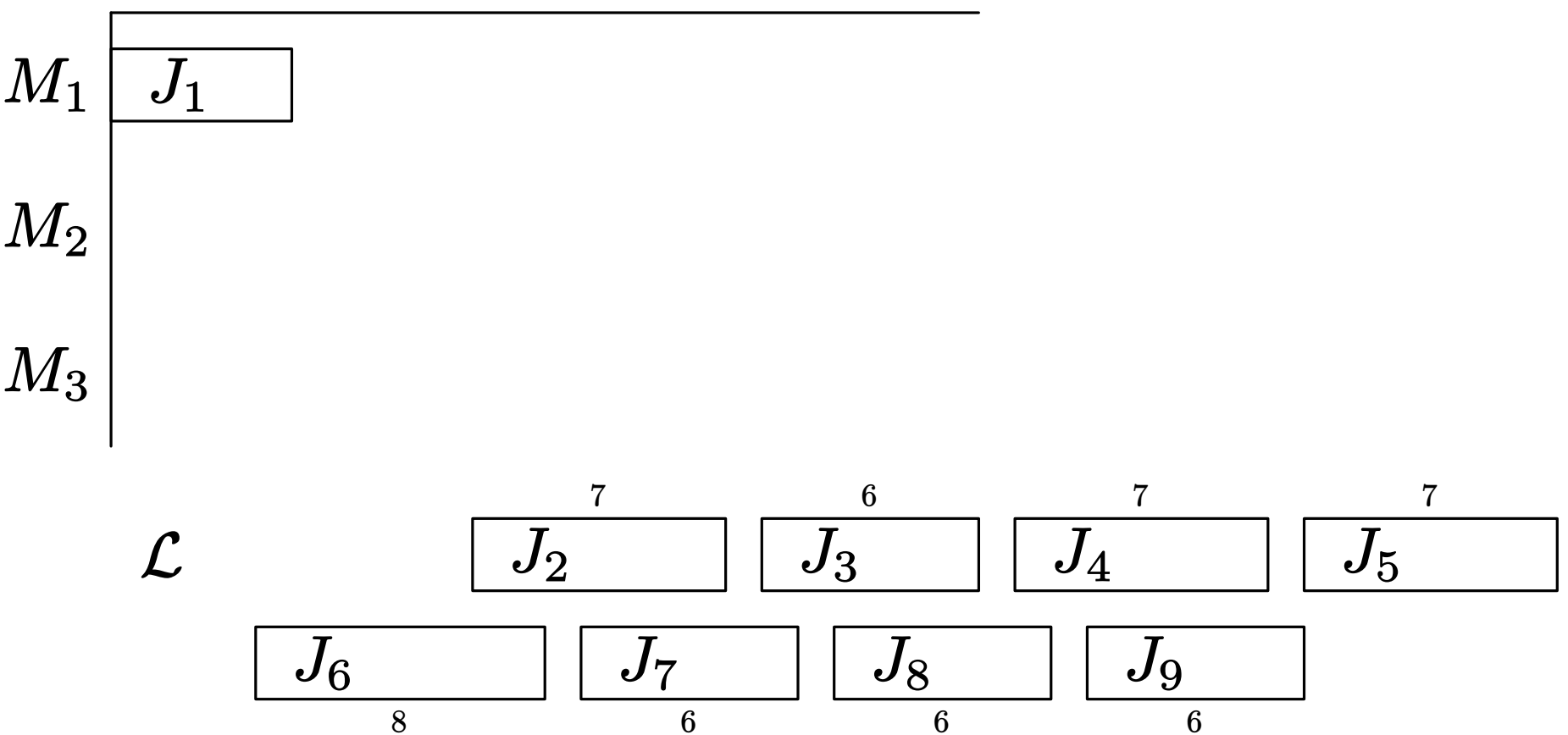
1. すべてのジョブを並べたリスト $\mathcal{L}$ を作る
2. 時刻を 0 から増やし，空いている機械があれば，
   それに $\mathcal{L}$ の先頭にあるジョブを割り当てる



リスト・スケジューリング：考え方

リスト・スケジューリング：ひな型

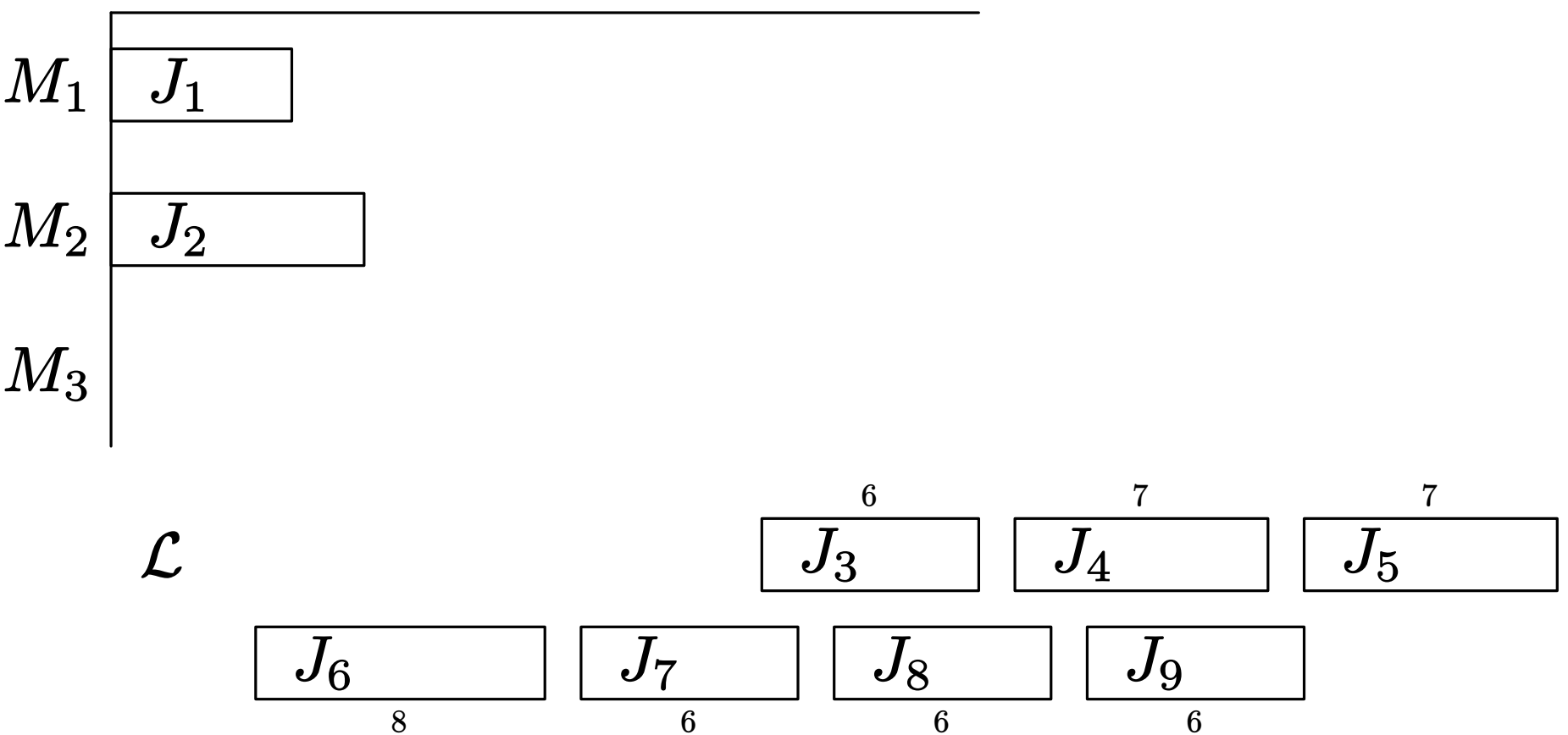
1. すべてのジョブを並べたリスト $\mathcal{L}$ を作る
2. 時刻を 0 から増やし，空いている機械があれば，
   それに $\mathcal{L}$ の先頭にあるジョブを割り当てる



リスト・スケジューリング：考え方

リスト・スケジューリング：ひな型

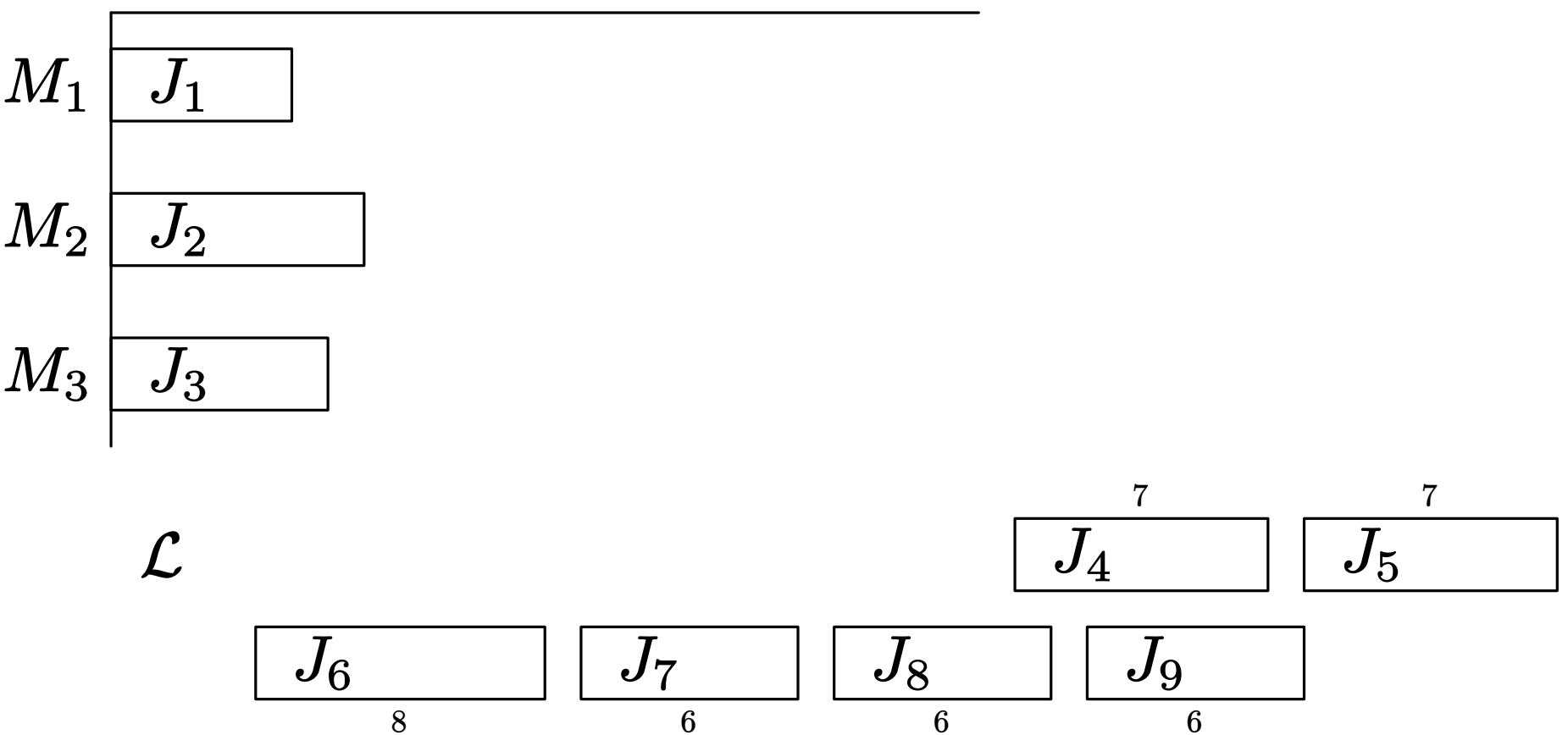
1. すべてのジョブを並べたリスト $\mathcal{L}$ を作る
2. 時刻を 0 から増やし，空いている機械があれば，
   それに $\mathcal{L}$ の先頭にあるジョブを割り当てる



リスト・スケジューリング：考え方

リスト・スケジューリング：ひな型

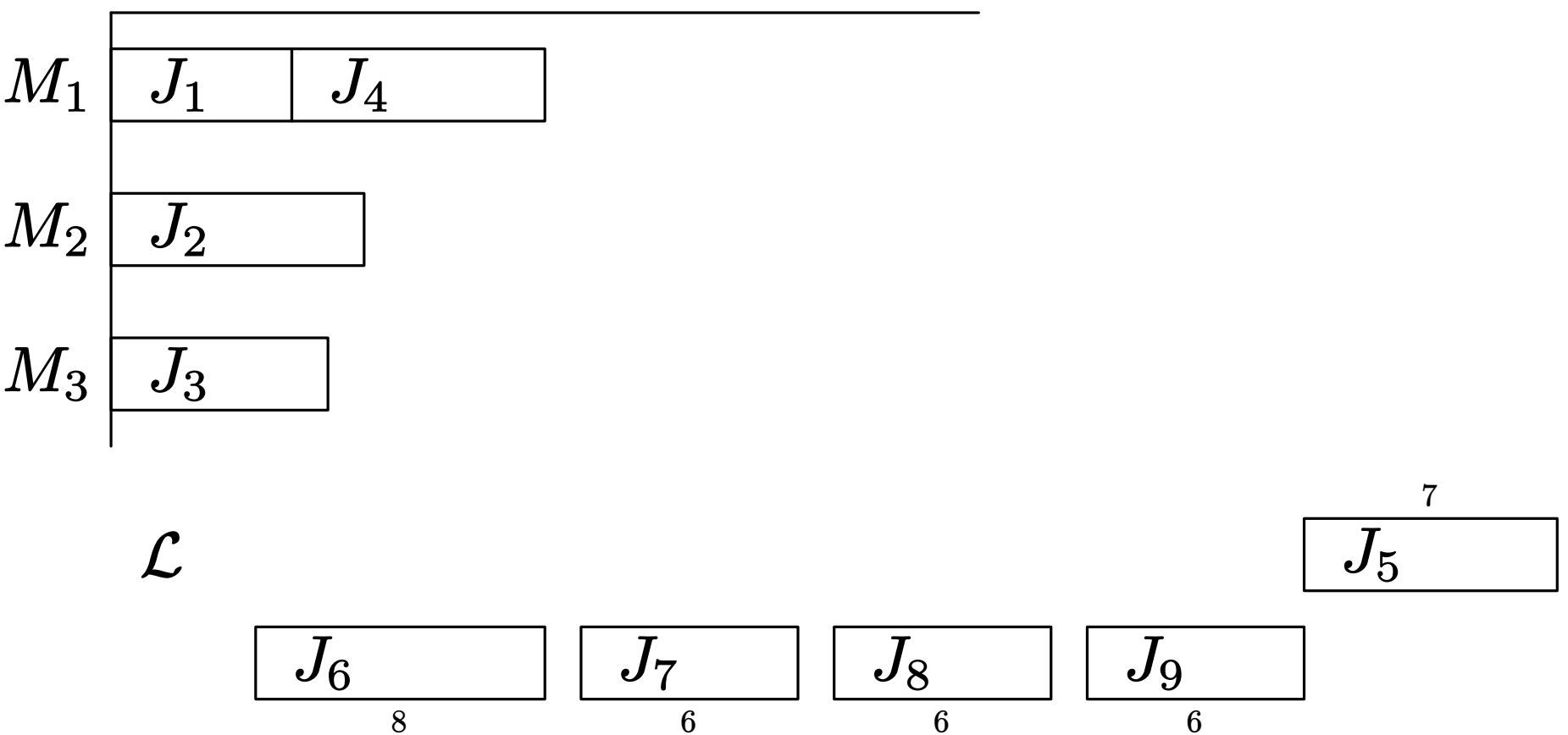
1. すべてのジョブを並べたリスト $\mathcal{L}$ を作る
2. 時刻を 0 から増やし，空いている機械があれば，
   それに $\mathcal{L}$ の先頭にあるジョブを割り当てる



リスト・スケジューリング：考え方

リスト・スケジューリング：ひな型

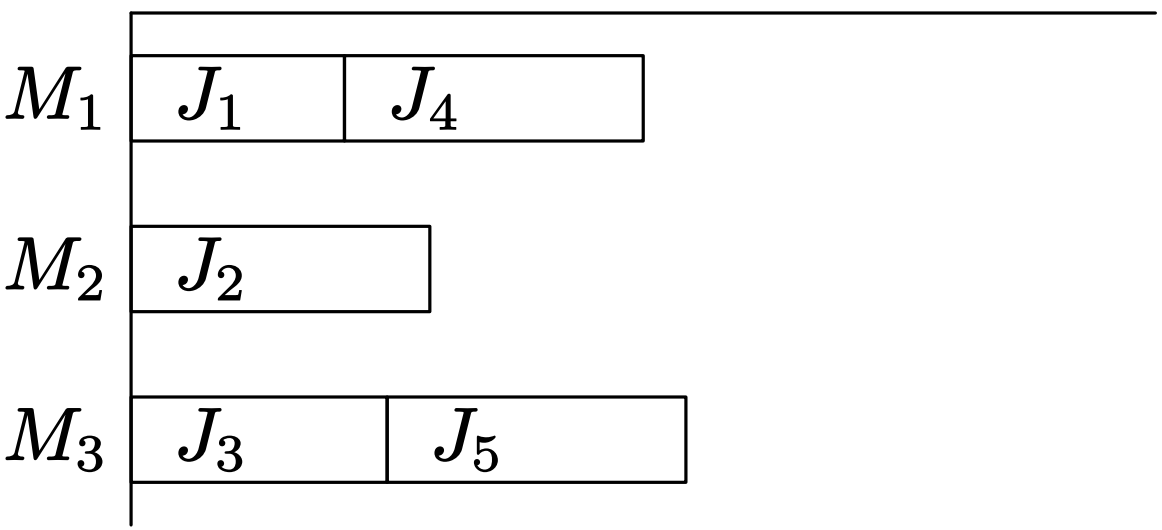
1. すべてのジョブを並べたリスト $\mathcal{L}$ を作る
2. 時刻を 0 から増やし，空いている機械があれば，
   それに $\mathcal{L}$ の先頭にあるジョブを割り当てる



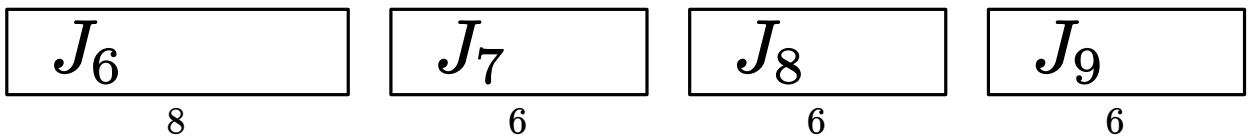
リスト・スケジューリング：考え方

リスト・スケジューリング：ひな型

1. すべてのジョブを並べたリスト $\mathcal{L}$ を作る
2. 時刻を 0 から増やし，空いている機械があれば，
   それに $\mathcal{L}$ の先頭にあるジョブを割り当てる



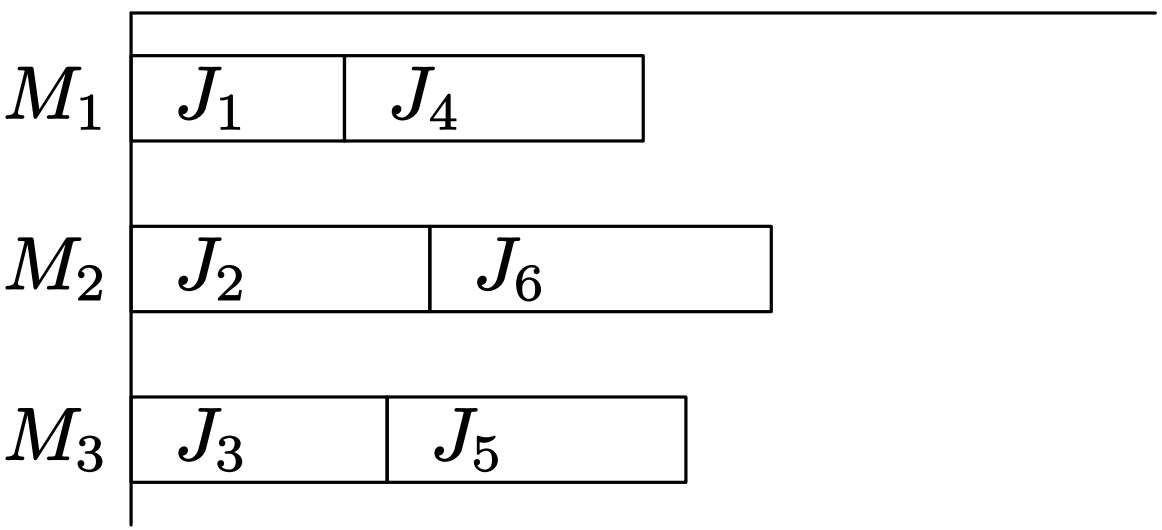
$\mathcal{L}$



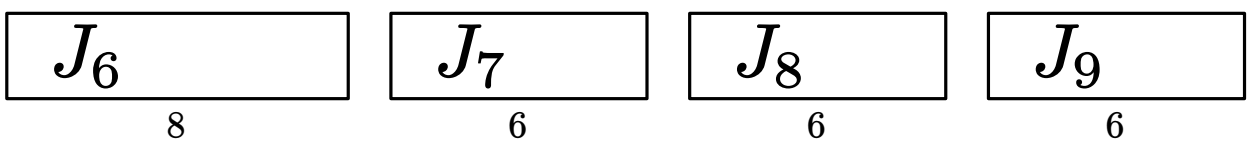
リスト・スケジューリング：考え方

リスト・スケジューリング：ひな型

1. すべてのジョブを並べたリスト $\mathcal{L}$ を作る
2. 時刻を 0 から増やし，空いている機械があれば，
   それに $\mathcal{L}$ の先頭にあるジョブを割り当てる



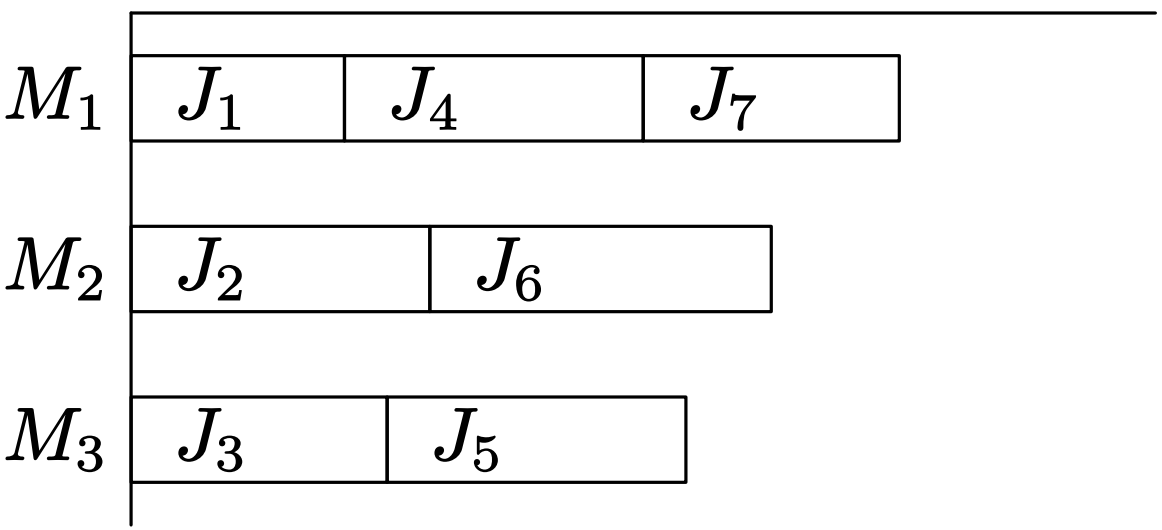
$\mathcal{L}$



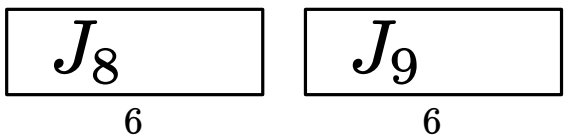
リスト・スケジューリング：考え方

リスト・スケジューリング：ひな型

1. すべてのジョブを並べたリスト $\mathcal{L}$ を作る
2. 時刻を 0 から増やし，空いている機械があれば，
   それに $\mathcal{L}$ の先頭にあるジョブを割り当てる



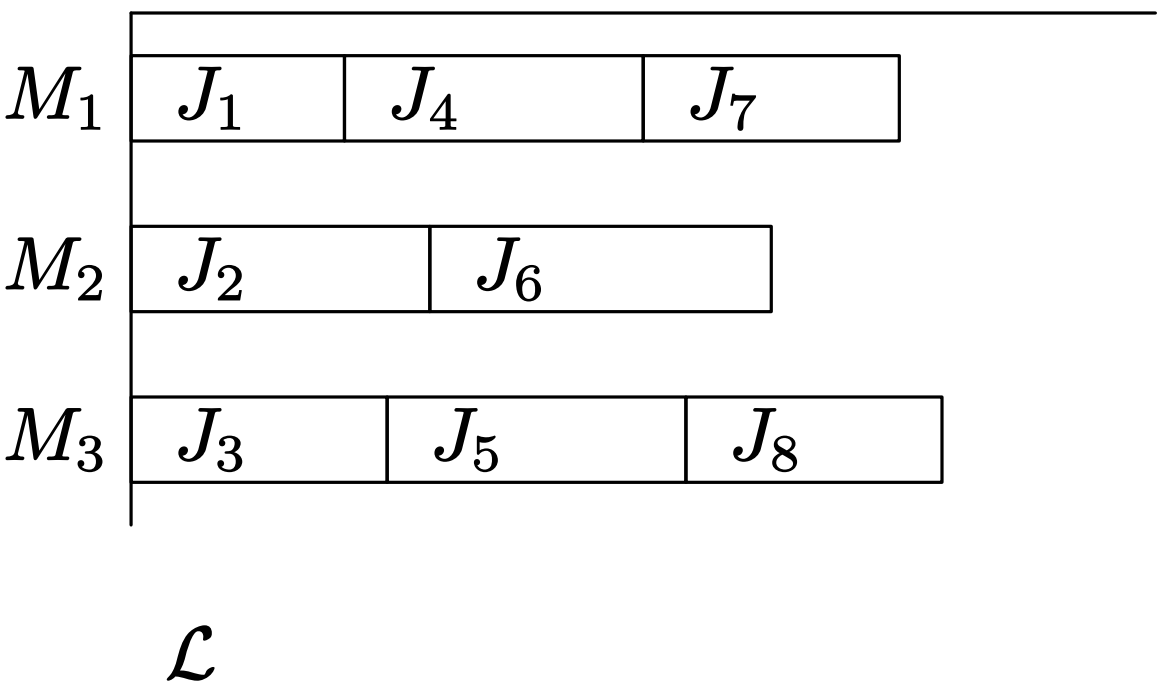
$\mathcal{L}$



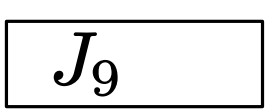
リスト・スケジューリング：考え方

リスト・スケジューリング：ひな型

1. すべてのジョブを並べたリスト $\mathcal{L}$ を作る
2. 時刻を 0 から増やし，空いている機械があれば，
   それに $\mathcal{L}$ の先頭にあるジョブを割り当てる



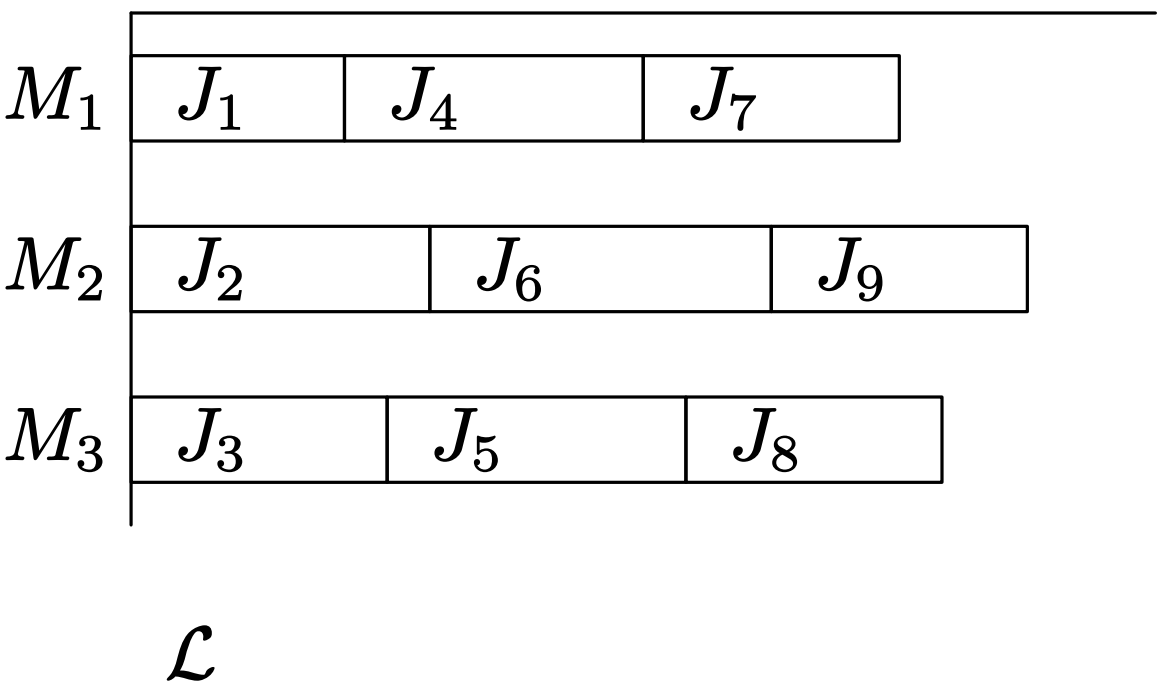
$\mathcal{L}$



リスト・スケジューリング：考え方

リスト・スケジューリング：ひな型

1. すべてのジョブを並べたリスト $\mathcal{L}$ を作る
2. 時刻を 0 から増やし，空いている機械があれば，
   それに $\mathcal{L}$ の先頭にあるジョブを割り当てる



$\mathcal{L}$

なにかしらの

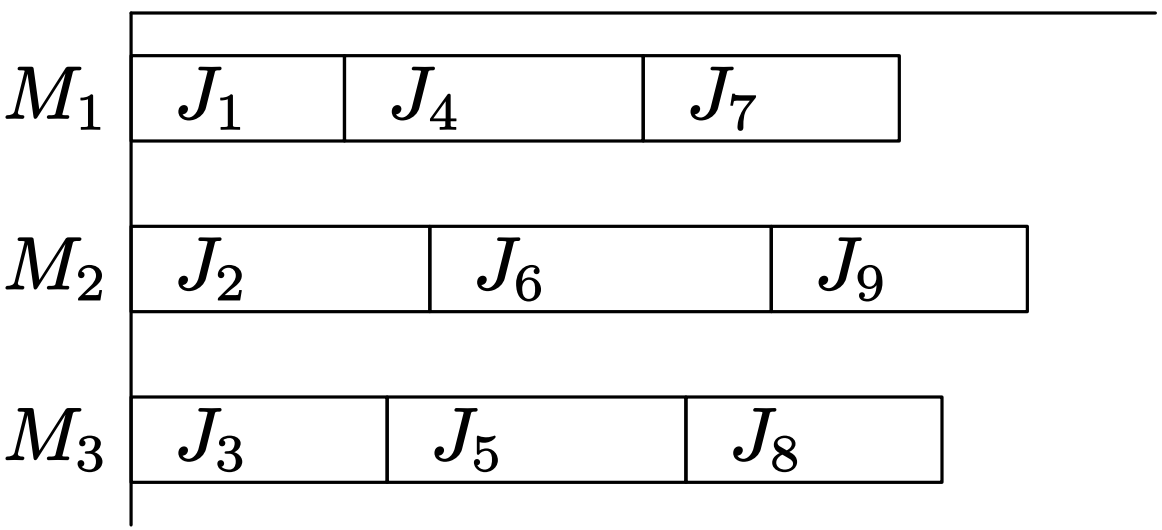
スケジュールが

得られる

リスト・スケジューリング：考え方

リスト・スケジューリング：ひな型

1. すべてのジョブを並べたリスト $\mathcal{L}$ を作る
2. 時刻を 0 から増やし，空いている機械があれば，
   それに $\mathcal{L}$ の先頭にあるジョブを割り当てる

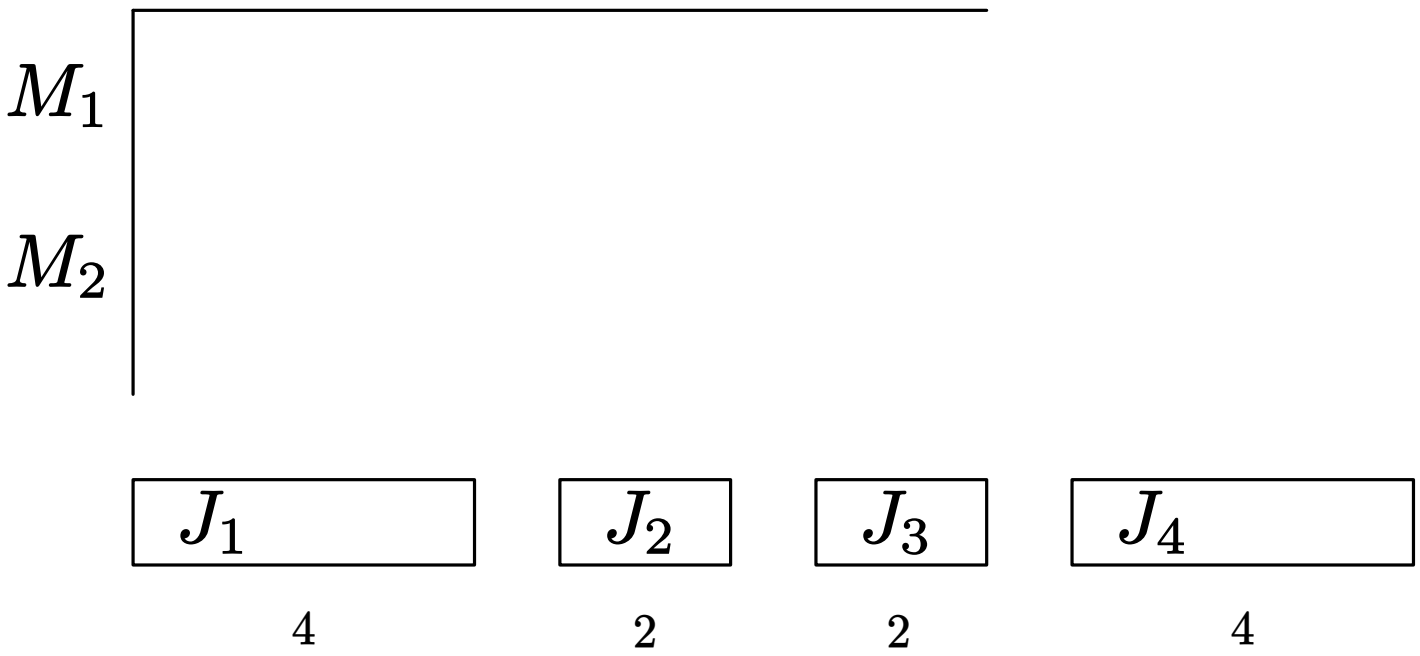


なにかしらの
スケジュールが
得られる

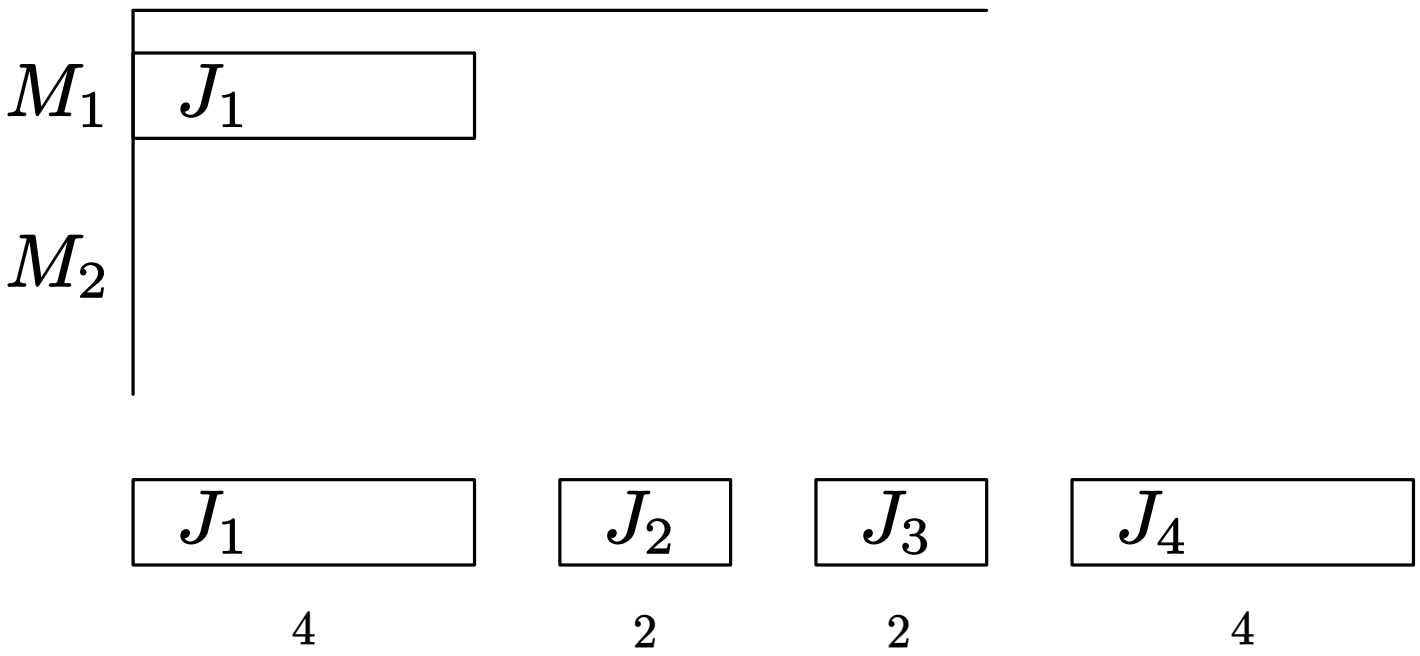
疑問

このスケジュールは **どれくらい良い** のか？

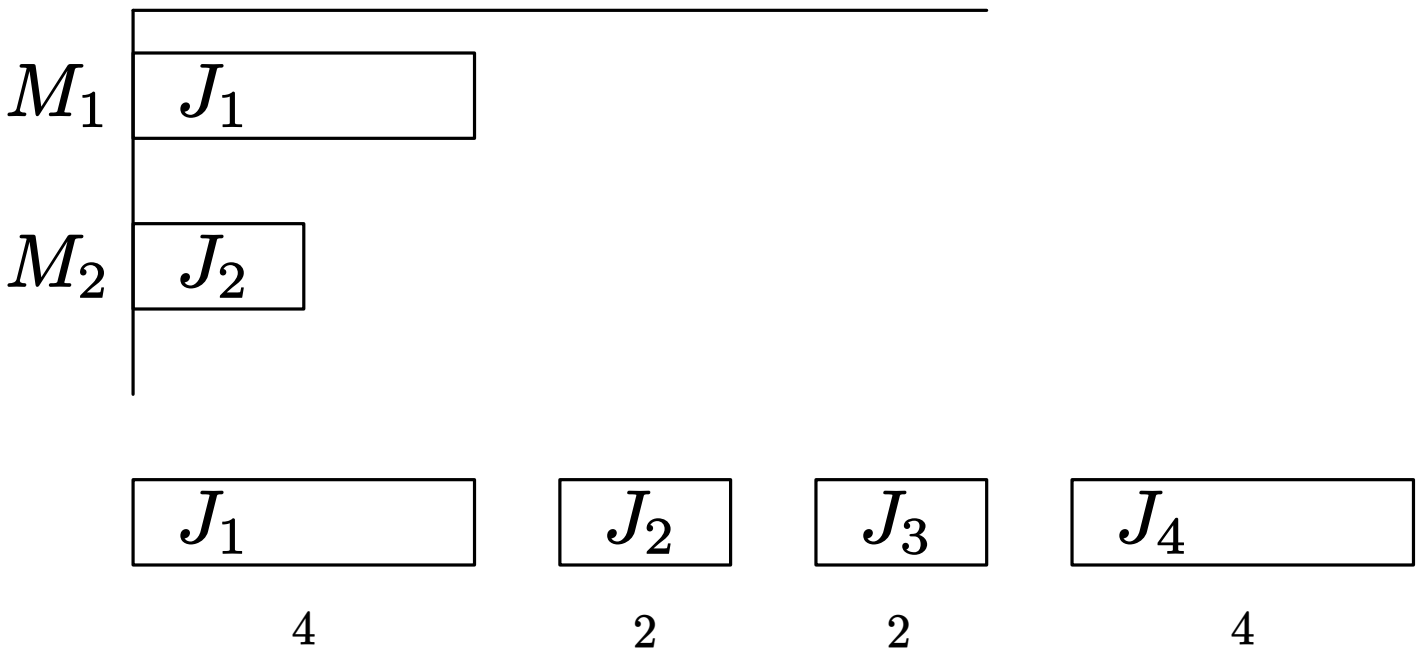
リスト・スケジューリングの悪さ



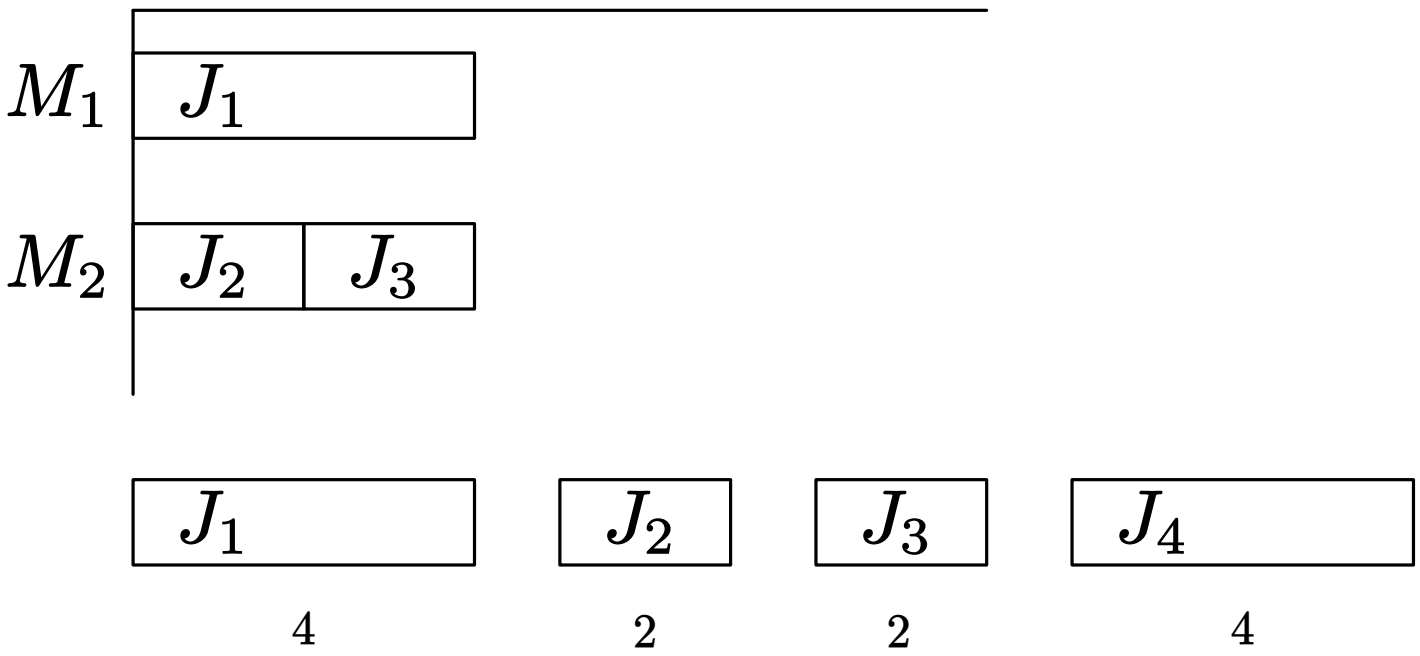
リスト・スケジューリングの悪さ



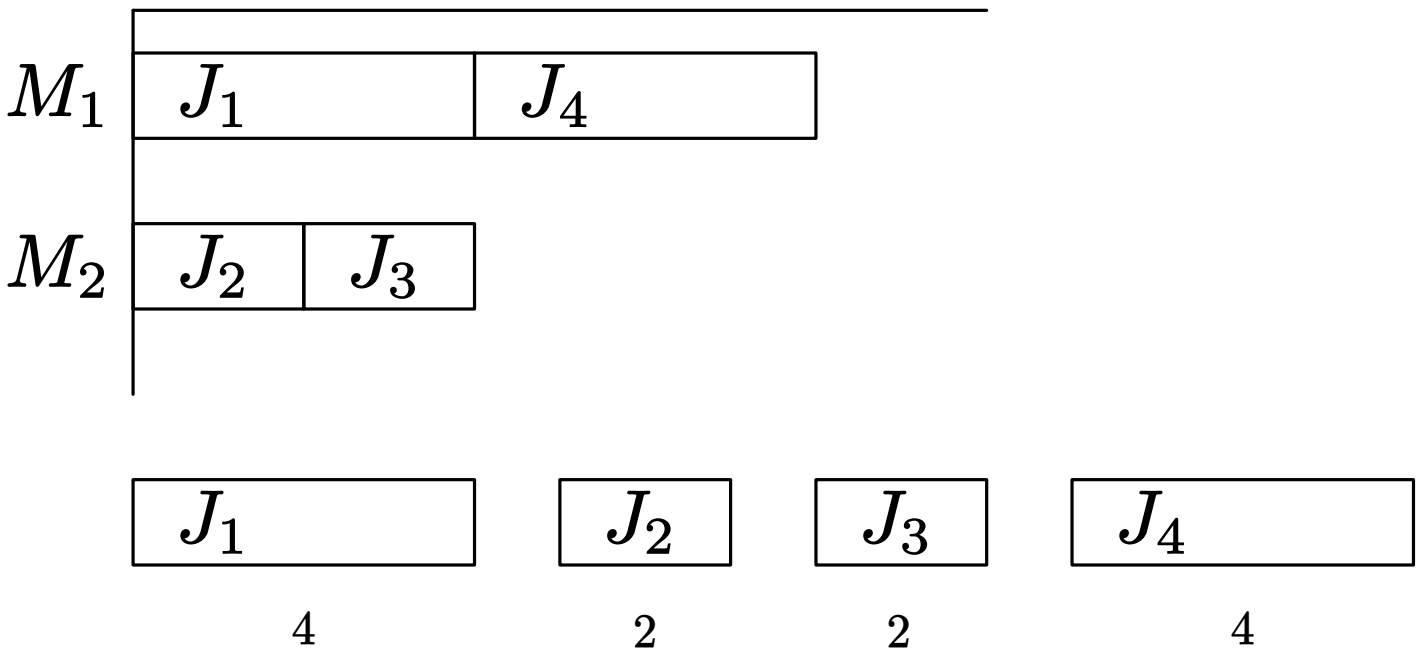
リスト・スケジューリングの悪さ



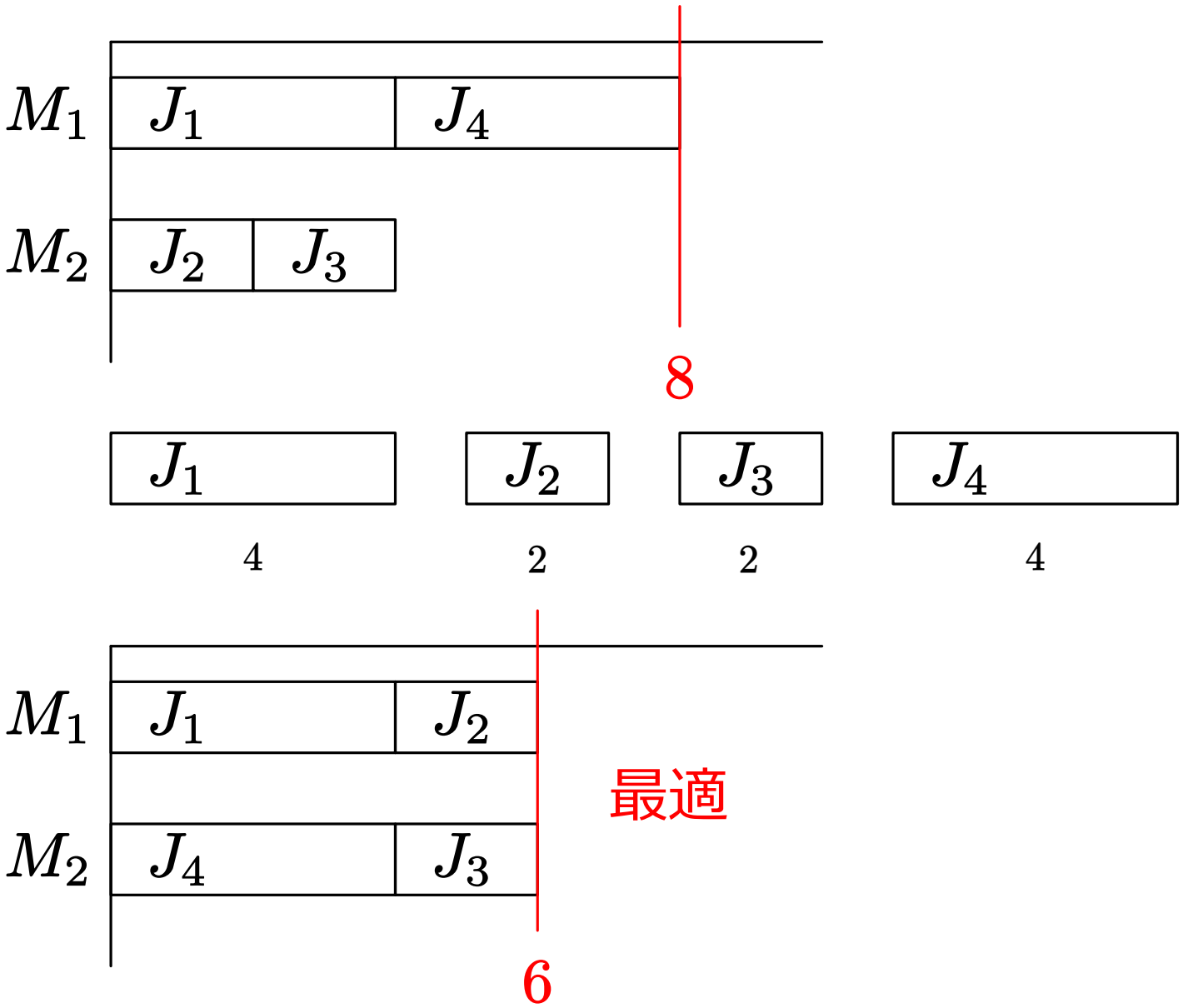
リスト・スケジューリングの悪さ



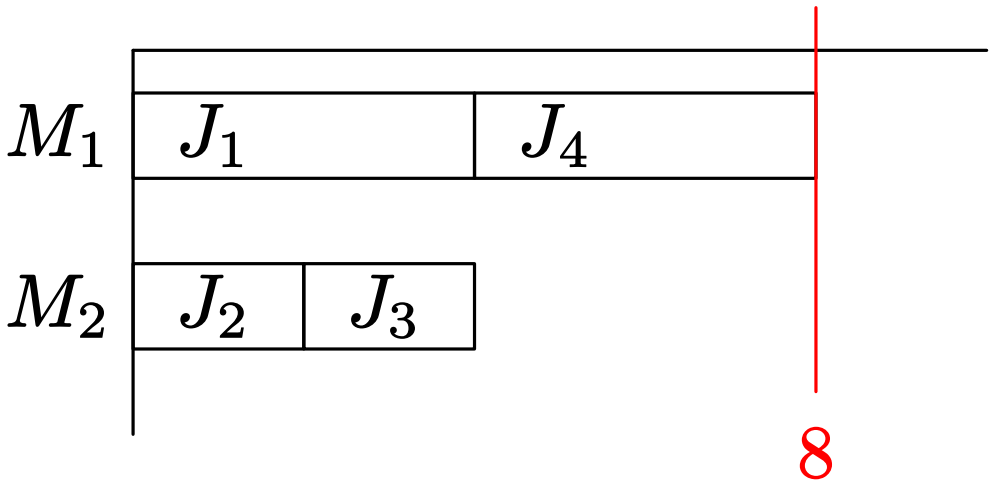
リスト・スケジューリングの悪さ



リスト・スケジューリングの悪さ

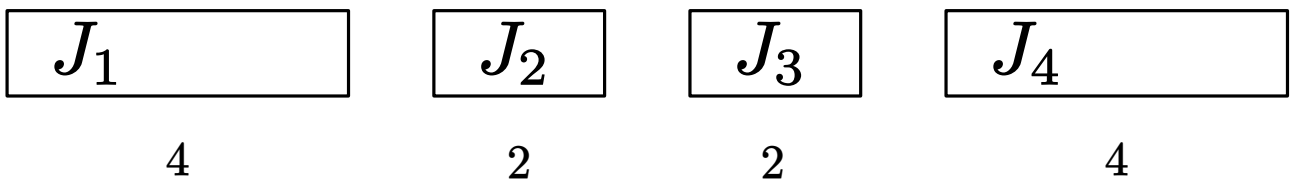


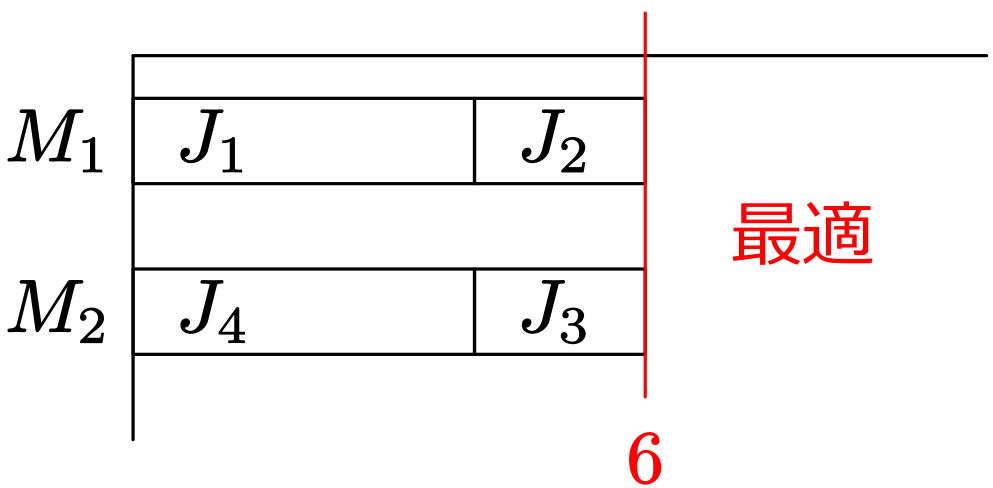
リスト・スケジューリングの悪さ



最適スケジュールと比べて

$\frac{8}{6} = \frac{4}{3}$ 倍しか悪くない





最適

α 近似

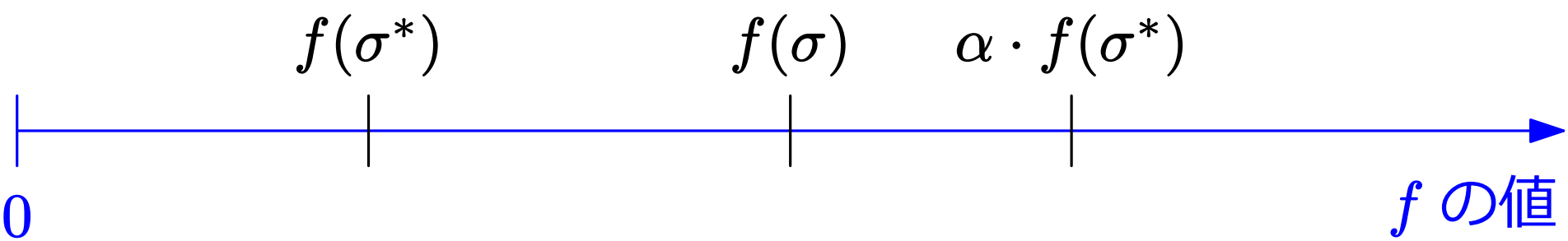
$\alpha \geq 1$ は実数

定義：α 近似 (α-approximation)

スケジュール σ が α **近似** であるとは，
最適スケジュール σ^* に対して

$$f(\sigma^*) \leq f(\sigma) \leq \alpha \cdot f(\sigma^*)$$

を満たすこと (ただし，f は目的関数)



別の表現：$1 \leq \dfrac{f(\sigma)}{f(\sigma^*)} \leq \alpha$　($f(\sigma^*) > 0$ のとき)

近似アルゴリズム

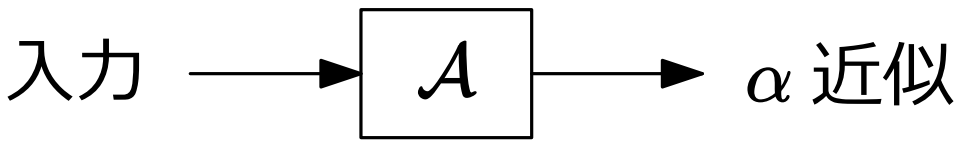
$\alpha \geq 1$ は実数

「アルゴリズム」の意味は後述

定義：α 近似アルゴリズム (α-approximation algorithm)

アルゴリズム $\mathcal{A}$ が α **近似** であるとは，
任意の入力に対して，$\mathcal{A}$ の出力が α 近似であること

気持ち：α が小さい $\Leftrightarrow$ よいアルゴリズム



近似アルゴリズム

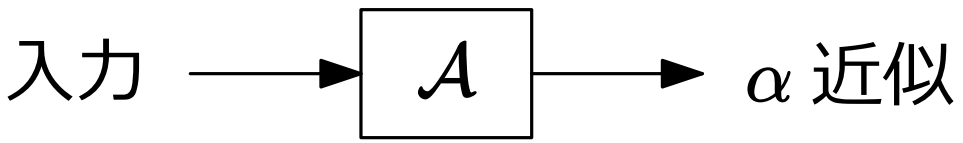
$\alpha \geq 1$ は実数

「アルゴリズム」の意味は後述

定義：α 近似アルゴリズム (α-approximation algorithm)

アルゴリズム $\mathcal{A}$ が α **近似** であるとは，
任意の入力に対して，$\mathcal{A}$ の出力が α 近似であること

気持ち：α が小さい $\Leftrightarrow$ よいアルゴリズム



用語：近似比，近似保証

この α を $\mathcal{A}$ の
- **近似比** (approximation ratio)
- **近似保証** (approximation guarantee)

と呼ぶことがある

リスト・スケジューリングの近似比

問題	リスト	近似比	
$1 \mid\mid C_{\max}$	任意	1	
$1 \mid\mid \sum C_j$	SPT	1	(Smith '56)
$1 \mid\mid L_{\max}$	EDD	1	(Jackson '55)
$1 \mid\mid \sum w_j C_j$	WSPT	1	(Smith '56)
$P \mid\mid C_{\max}$	任意	$2 - \frac{1}{m}$	(Graham '66)
	LPT	$\frac{4}{3} - \frac{1}{3m}$	(Graham '69)
$P \mid\mid \sum C_j$	SPT	1	(Conway, Maxwell, Miller '67)
$P \mid\mid \sum w_j C_j$	WSPT	$\frac{1}{2}(1+\sqrt{2})$	(Kawaguchi, Kyan '86)

リスト・スケジューリングの近似比

問題	リスト	近似比	
$1 \mid\mid C_{\max}$	任意	1	
$1 \mid\mid \sum C_j$	SPT	1	(Smith '56)
$1 \mid\mid L_{\max}$	EDD	1	(Jackson '55)
$1 \mid\mid \sum w_j C_j$	WSPT	1	(Smith '56)
$P \mid\mid C_{\max}$	任意	$2 - \frac{1}{m}$	(Graham '66)
	LPT	$\frac{4}{3} - \frac{1}{3m}$	(Graham '69)
$P \mid\mid \sum C_j$	SPT	1	(Conway, Maxwell, Miller '67)
$P \mid\mid \sum w_j C_j$	WSPT	$\frac{1}{2}(1+\sqrt{2})$	(Kawaguchi, Kyan '86)

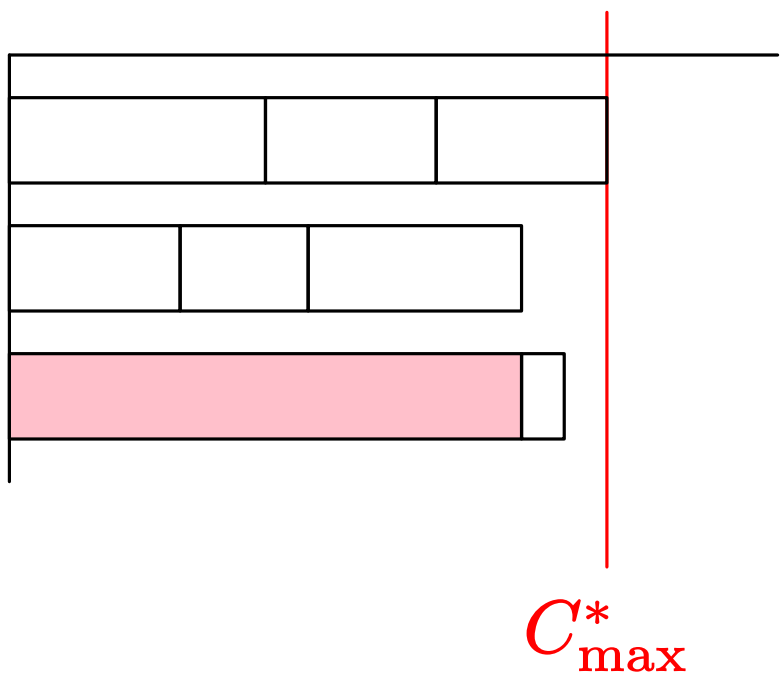
$P \mid\mid C_{\max}$：任意のリスト

定理：$P \mid\mid C_{\max}$ と任意リスト (Graham '66)

$P \mid\mid C_{\max}$ において，
任意のリストに対するリスト・スケジューリングは
$\left(2 - \frac{1}{m}\right)$ 近似アルゴリズムである (m は機械数)

証明：任意の入力を考えて，最適値を $C^*_{\max}$ とする

- $C^*_{\max}$ は次を満たす
  - $C^*_{\max} \geq p_j \ (\forall\, j \in [n])$
  - $C^*_{\max} \geq \frac{1}{m} \sum_{j=1}^{n} p_j$



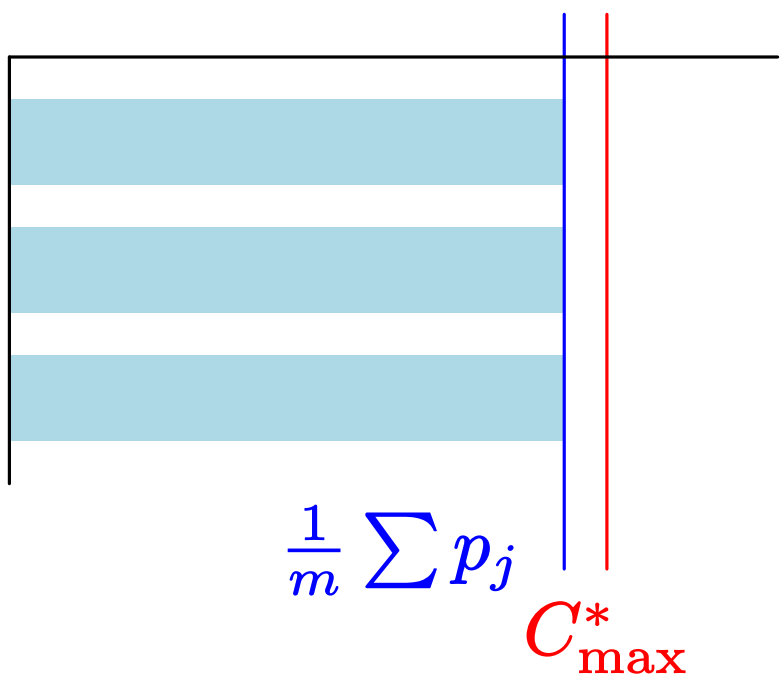
$P \mid\mid C_{\max}$：任意のリスト

定理：$P \mid\mid C_{\max}$ と任意リスト (Graham '66)

$P \mid\mid C_{\max}$ において，
任意のリストに対するリスト・スケジューリングは
$\left(2 - \frac{1}{m}\right)$ 近似アルゴリズムである (m は機械数)

証明：任意の入力を考えて，最適値を $C^*_{\max}$ とする

- $C^*_{\max}$ は次を満たす
  - $C^*_{\max} \geq p_j \ (\forall\, j \in [n])$
  - $C^*_{\max} \geq \frac{1}{m} \sum_{j=1}^{n} p_j$



$P \mid\mid C_{\max}$：任意のリスト

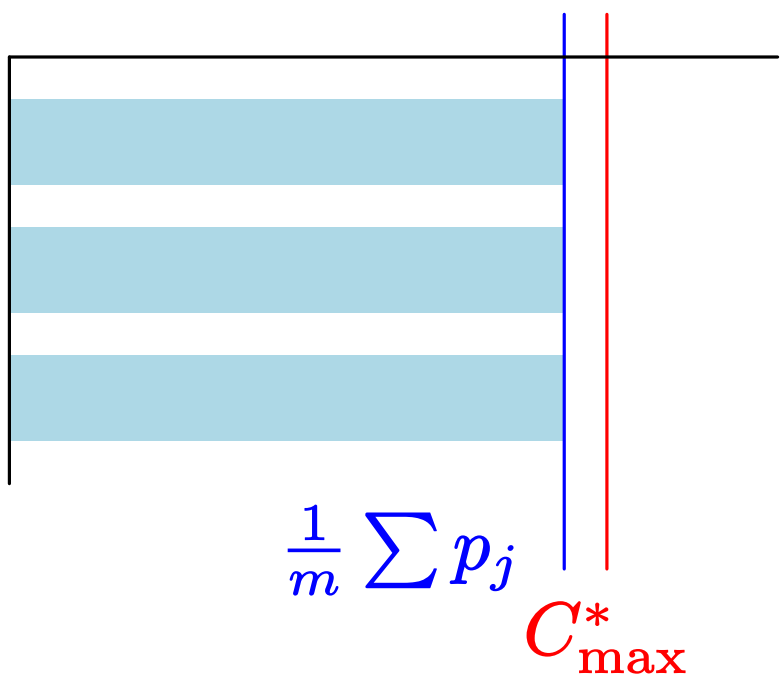
定理：$P \mid\mid C_{\max}$ と任意リスト (Graham '66)

$P \mid\mid C_{\max}$ において，
任意のリストに対するリスト・スケジューリングは
$\left(2 - \frac{1}{m}\right)$ 近似アルゴリズムである (m は機械数)

証明：任意の入力を考えて，最適値を $C^*_{\max}$ とする

- $C^*_{\max}$ は次を満たす
  - $C^*_{\max} \geq p_j \ (\forall\, j \in [n])$
  - $C^*_{\max} \geq \frac{1}{m} \sum_{j=1}^{n} p_j$

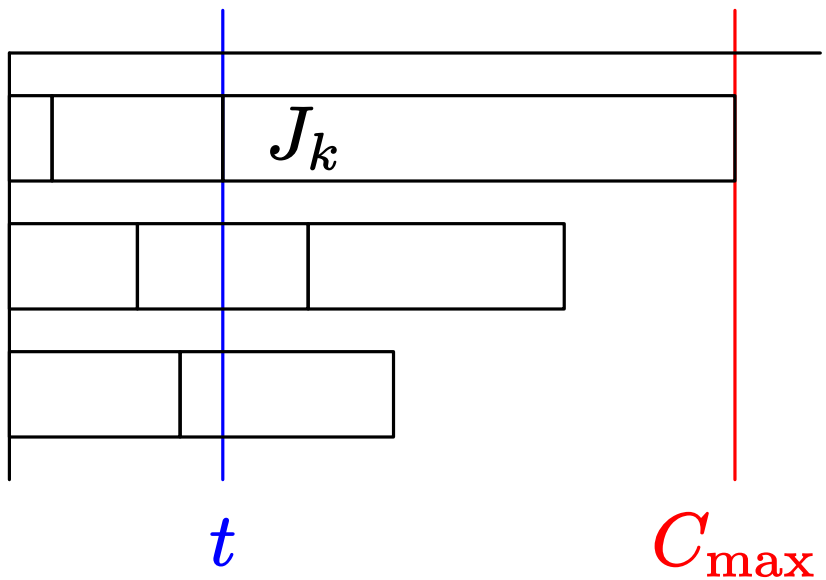
最適値の下界



$P \mid\mid C_{\max}$：任意のリスト (続)

証明 (続き)：アルゴリズムの出力値を $C_{\max}$ とする

- 完了時刻が $C_{\max}$ である機械で最後に処理されるジョブを J_k として，その処理開始時刻を t とする

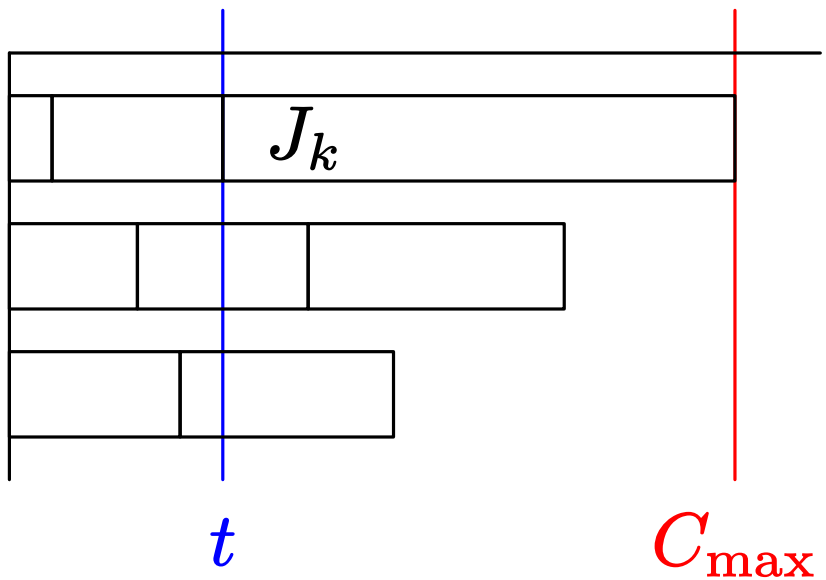


$P \mid\mid C_{\max}$：任意のリスト (続)

証明 (続き)：アルゴリズムの出力値を $C_{\max}$ とする

- 完了時刻が $C_{\max}$ である機械で最後に処理されるジョブを J_k として，その処理開始時刻を t とする
- $C_{\max} = t + p_k$
- t の直前まで，すべての機械がジョブを処理しているので

$$t \leq \frac{1}{m}\left(\sum_{j=1}^{n} p_j - p_k\right)$$

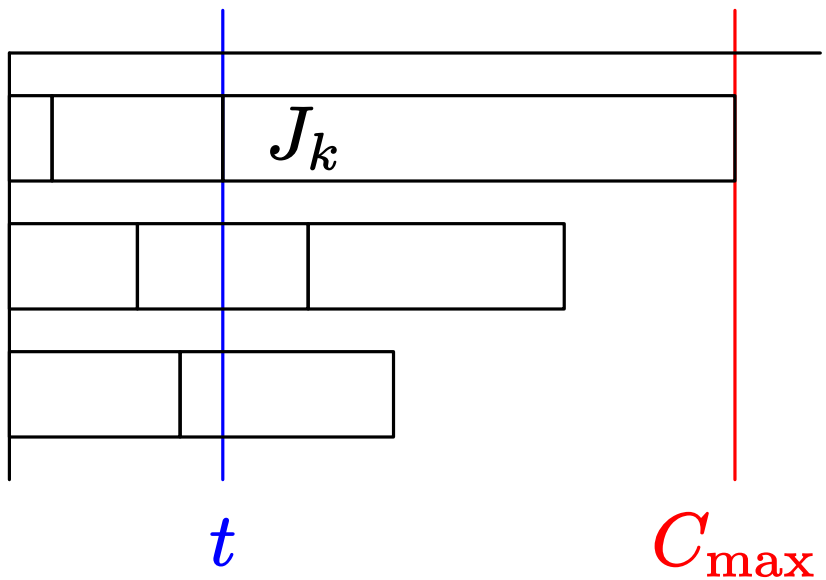


$P \mid\mid C_{\max}$：任意のリスト (続)

証明 (続き)：アルゴリズムの出力値を $C_{\max}$ とする

- 完了時刻が $C_{\max}$ である機械で最後に処理されるジョブを J_k として，その処理開始時刻を t とする
- $C_{\max} = t + p_k$
- t の直前まで，すべての機械がジョブを処理しているので

$$t \leq \frac{1}{m}\left(\sum_{j=1}^{n} p_j - p_k\right)$$



- したがって，

$$\begin{aligned} C_{\max} &\leq \frac{1}{m}\left(\sum p_j - p_k\right) + p_k \\ &= \frac{1}{m}\sum p_j + \left(1 - \frac{1}{m}\right) p_k \\ &\leq C^*_{\max} + \left(1 - \frac{1}{m}\right) C^*_{\max} = \left(2 - \frac{1}{m}\right) C^*_{\max} \end{aligned}$$

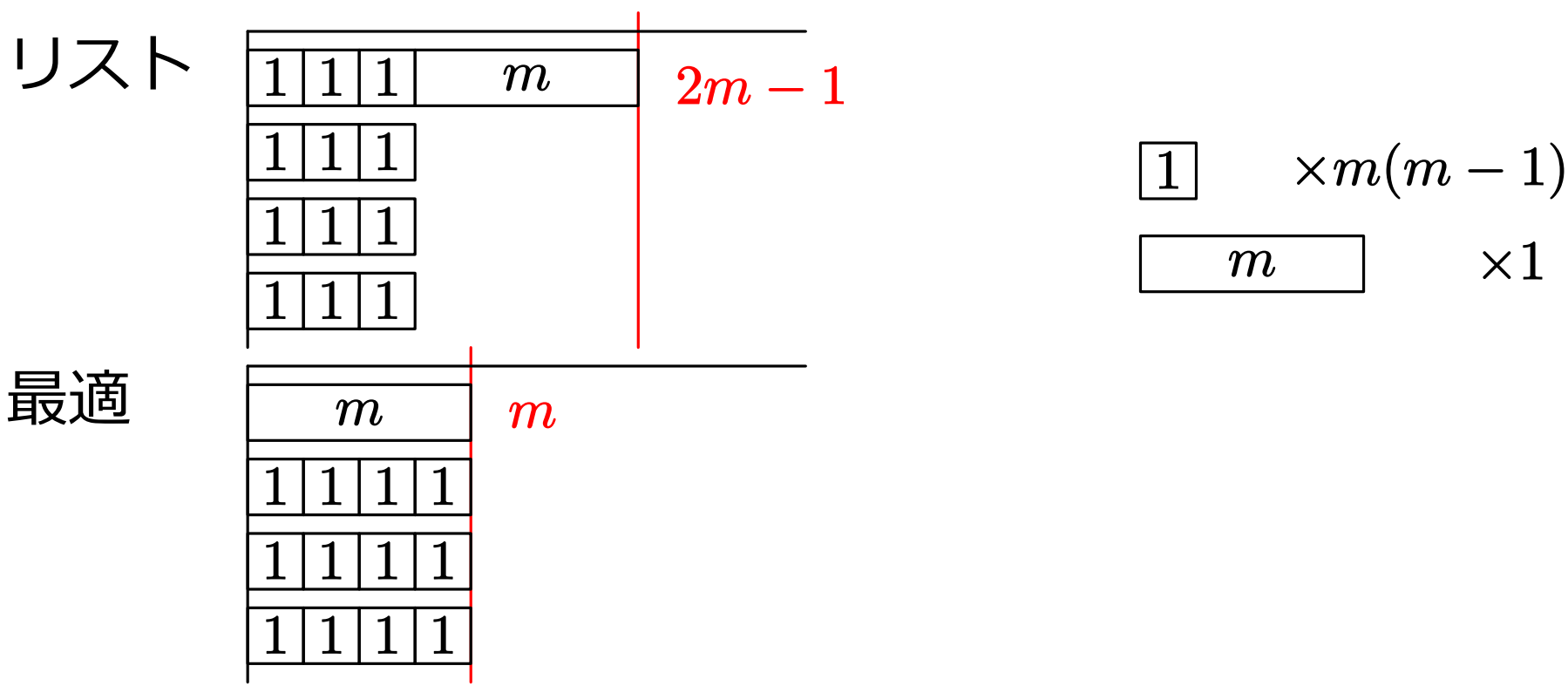
□

$P \mid\mid C_{\max}$：任意のリスト(タイト)

性質：近似比がタイト

どの整数 $m \geq 1$ に対しても，
任意のリストに対するリスト・スケジューリングの
近似比は $2 - \frac{1}{m}$ より小さくない

この意味で，$2 - \frac{1}{m}$ という近似比は **タイト** (tight) である



リスト・スケジューリングの近似比

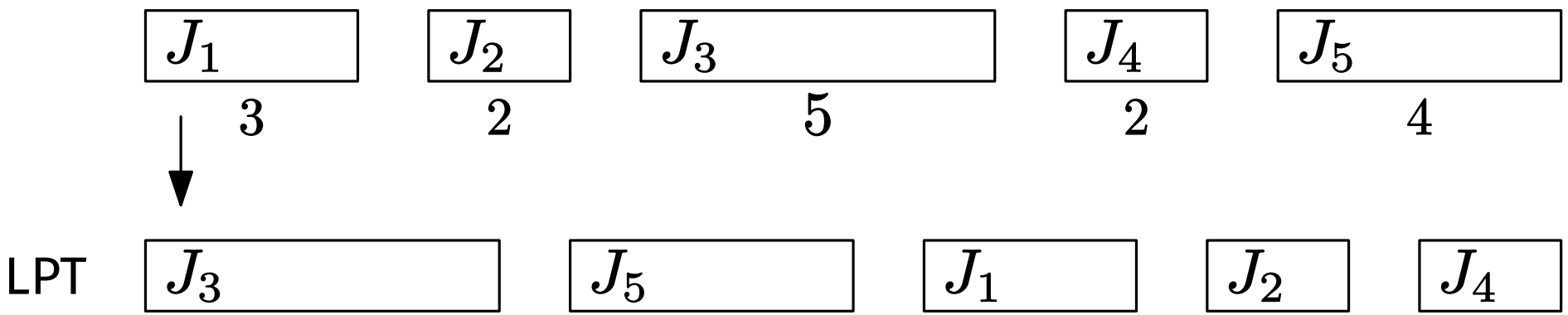
問題	リスト	近似比	
$1 \mid\mid C_{\max}$	任意	1	
$1 \mid\mid \sum C_j$	SPT	1	(Smith '56)
$1 \mid\mid L_{\max}$	EDD	1	(Jackson '55)
$1 \mid\mid \sum w_j C_j$	WSPT	1	(Smith '56)
$P \mid\mid C_{\max}$	任意	$2 - \frac{1}{m}$	(Graham '66)
	LPT	$\frac{4}{3} - \frac{1}{3m}$	(Graham '69)
$P \mid\mid \sum C_j$	SPT	1	(Conway, Maxwell, Miller '67)
$P \mid\mid \sum w_j C_j$	WSPT	$\frac{1}{2}(1+\sqrt{2})$	(Kawaguchi, Kyan '86)

最長処理時間優先順 (LPT)

リストの作り方：最長処理時間優先順 (LPT)

処理時間が長い順にジョブを並べる

LPT = Longest Processing Time

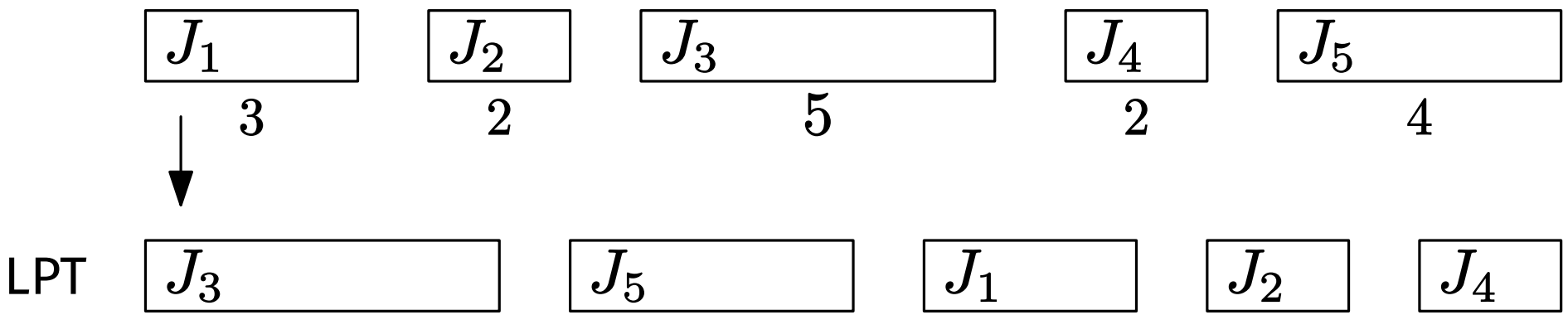


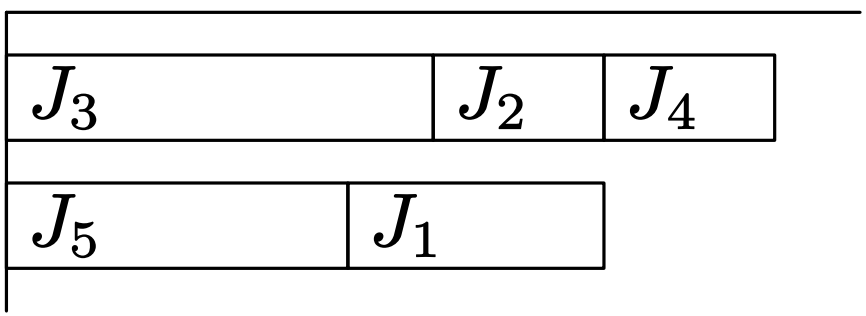
最長処理時間優先順 (LPT)

リストの作り方：最長処理時間優先順 (LPT)

処理時間が長い順にジョブを並べる

LPT = Longest Processing Time





$P \mid\mid C_{\max}$ と LPT (1)

定理：$P \mid\mid C_{\max}$ と LPT (Graham '69)

$P \mid\mid C_{\max}$ において，
LPT に対するリスト・スケジューリングは
$\left(\frac{4}{3} - \frac{1}{3m}\right)$ 近似アルゴリズムである　(m は機械数)

証明：任意の入力を考えて，最適値を $C^*_{\max}$ とする
アルゴリズムの出力値を $C_{\max}$ とする

次を順番に証明する

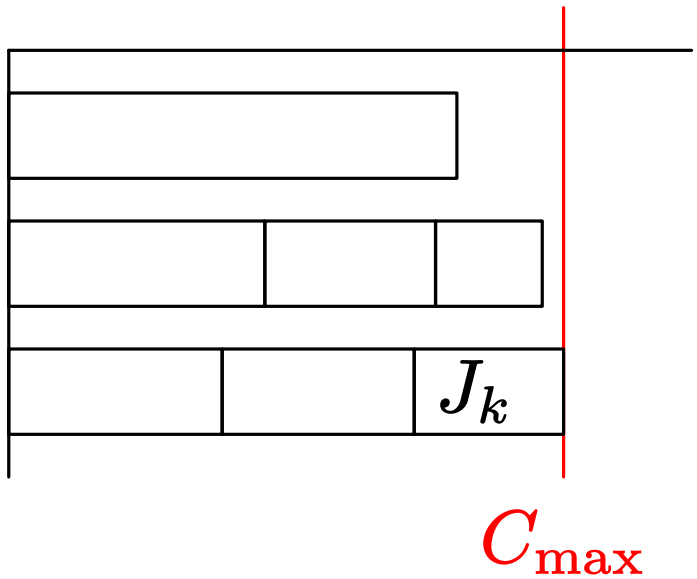
1. 最後に完了するジョブの処理時間が一番短いとして
   一般性を失わない
2. すべての $j \in [n]$ で $p_j > C^*_{\max}/3 \Rightarrow C_{\max} = C^*_{\max}$

$P \mid\mid C_{\max}$ と LPT：補題 1

- LPT で最後に処理されるジョブを J_k とする

補題 1

$p_k = \min\{p_1, \ldots, p_n\}$ の場合に定理が証明できれば十分



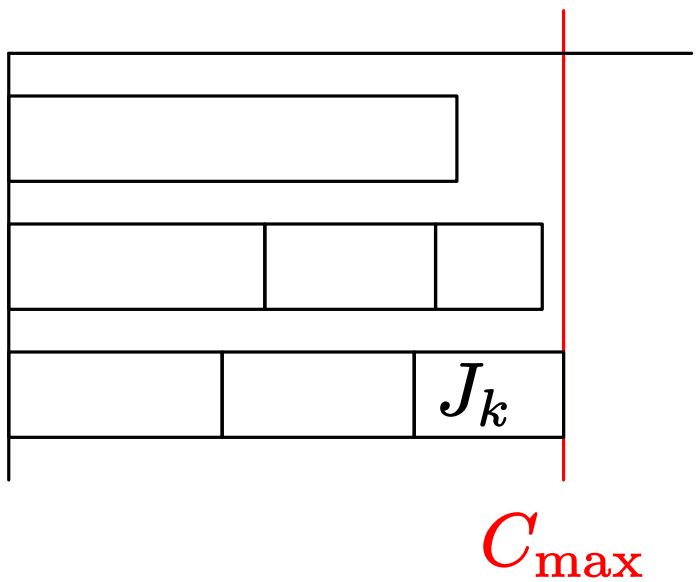
$P \mid\mid C_{\max}$ と LPT：補題 1

- LPT で最後に処理されるジョブを J_k とする

補題 1

$p_k = \min\{p_1, \ldots, p_n\}$ の場合に定理が証明できれば十分

- J_k よりも処理時間が短いジョブは
  J_k よりも早く処理開始されない
- それが無くても
  LPT による最大完了時刻や
  最後に処理するジョブは変わらない



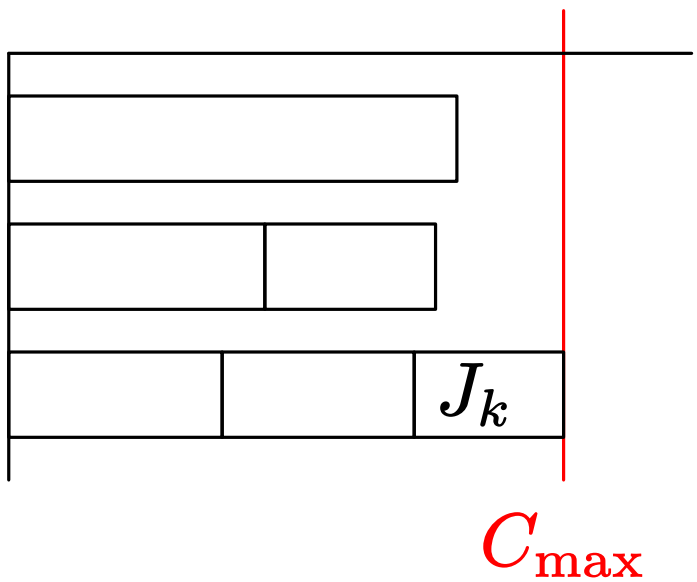
$P \mid\mid C_{\max}$ と LPT：補題 1

- LPT で最後に処理されるジョブを J_k とする

補題 1

$p_k = \min\{p_1, \ldots, p_n\}$ の場合に定理が証明できれば十分

- J_k よりも処理時間が短いジョブは
  J_k よりも早く処理開始されない
- それが無くても
  LPT による最大完了時刻や
  最後に処理するジョブは変わらない



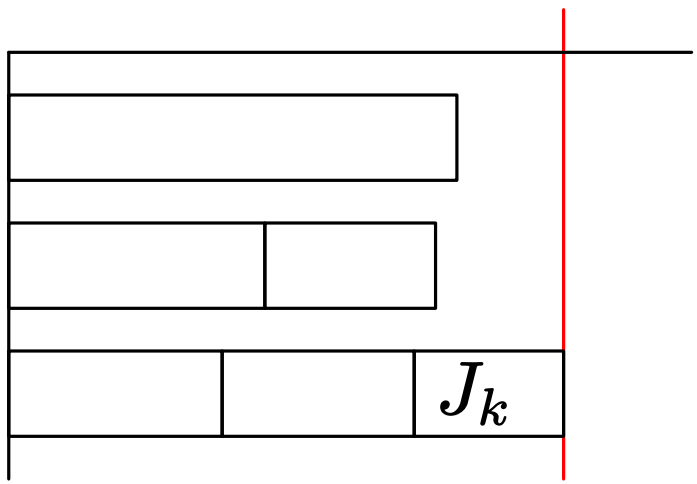
$P \mid\mid C_{\max}$ と LPT：補題 1

- LPT で最後に処理されるジョブを J_k とする

補題 1

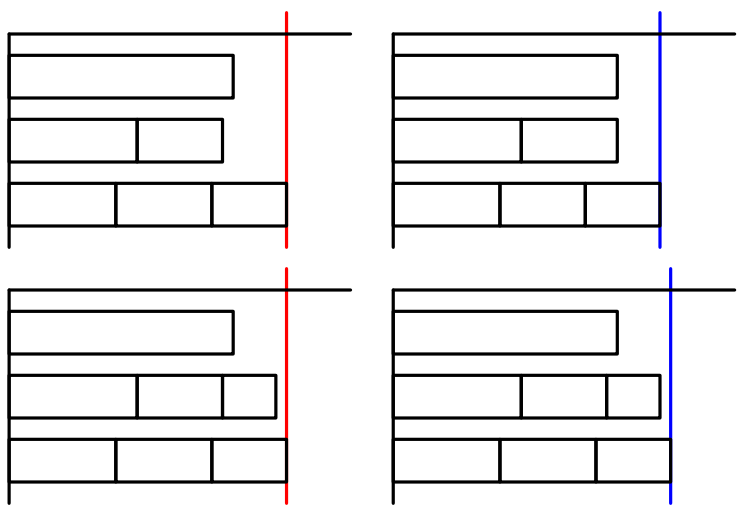
$p_k = \min\{p_1, \ldots, p_n\}$ の場合に定理が証明できれば十分

- J_k よりも処理時間が短いジョブは J_k よりも早く処理開始されない
- それが無くても LPT による最大完了時刻や最後に処理するジョブは変わらない



削除後

削除前



$$C'_{\max} \leq \alpha \cdot C'^{*}_{\max}$$

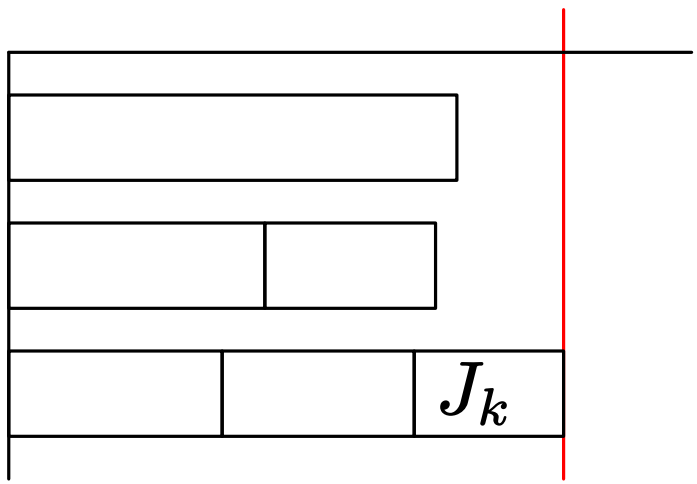
$P \mid\mid C_{\max}$ と LPT：補題 1

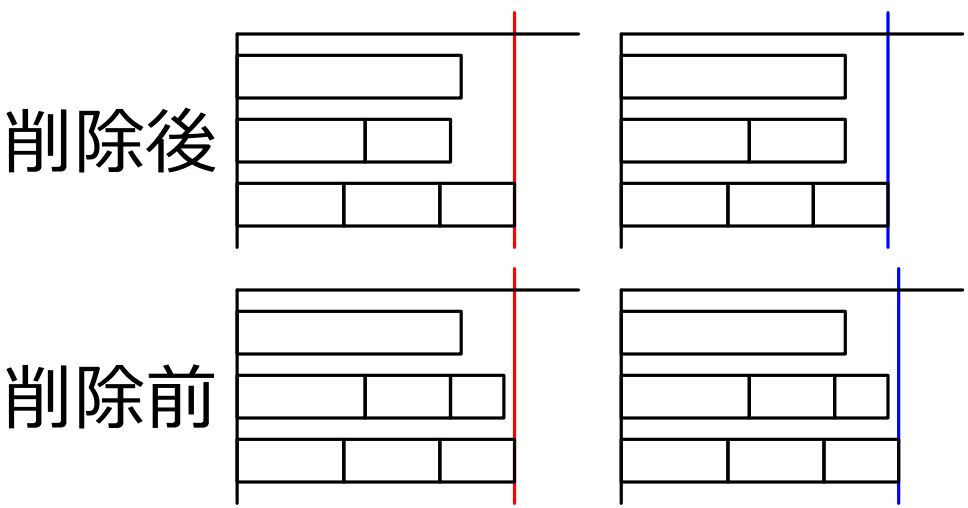
- LPT で最後に処理されるジョブを J_k とする

補題 1

$p_k = \min\{p_1, \ldots, p_n\}$ の場合に定理が証明できれば十分

- J_k よりも処理時間が短いジョブは J_k よりも早く処理開始されない
- それが無くても LPT による最大完了時刻や最後に処理するジョブは変わらない





$$\underbrace{C'_{\max}}_{=C_{\max}} \leq \alpha \cdot \underbrace{C'^{*}_{\max}}_{\leq C^{*}_{\max}}$$

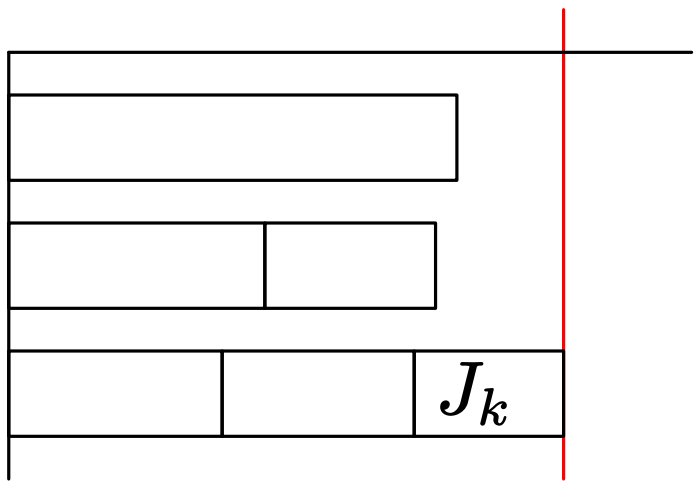
$P \mid\mid C_{\max}$ と LPT：補題 1

- LPT で最後に処理されるジョブを J_k とする

補題 1

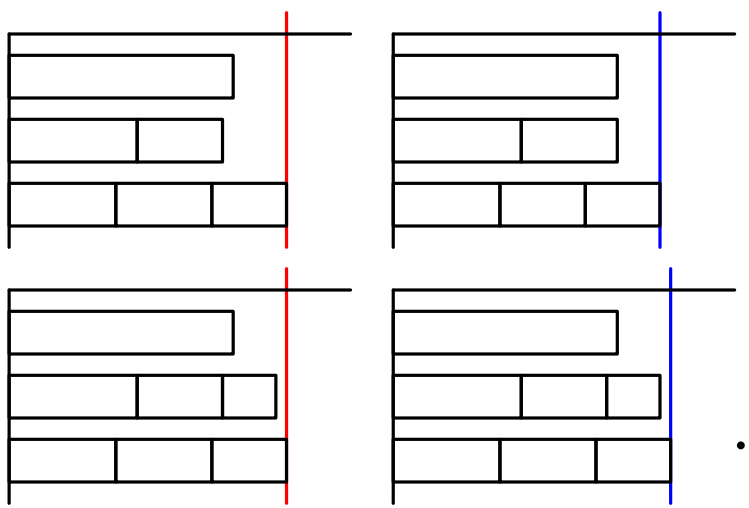
$p_k = \min\{p_1, \ldots, p_n\}$ の場合に定理が証明できれば十分

- J_k よりも処理時間が短いジョブは J_k よりも早く処理開始されない
- それが無くても LPT による最大完了時刻や最後に処理するジョブは変わらない



削除後

削除前



$$\underline{C'_{\max}} \leq \alpha \cdot \underline{C'^{*}_{\max}}$$

$$= C_{\max} \qquad \leq C^{*}_{\max}$$

$$\therefore\ C_{\max} \leq \alpha \cdot C^{*}_{\max}$$

□

$P \mid\mid C_{\max}$ と LPT：補題 2

- LPT 順を $J_1, J_2, \ldots, J_n$ とする $(p_1 \geq p_2 \geq \cdots \geq p_n)$

補題 2

$p_n > C^*_{\max}/3 \Rightarrow C_{\max} = C^*_{\max}$

$P \mid\mid C_{\max}$ と LPT：補題 2

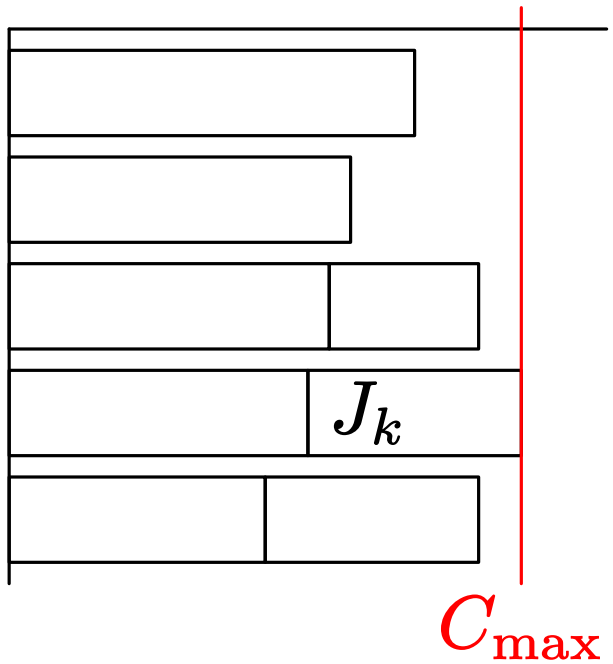
- LPT 順を $J_1, J_2, \ldots, J_n$ とする $(p_1 \geq p_2 \geq \cdots \geq p_n)$

補題 2

$p_n > C^*_{\max}/3 \Rightarrow C_{\max} = C^*_{\max}$

この仮定の下，最適解で各機械が処理するジョブ数 ≤ 2

- このとき，LPT が最適解を与える □



$P \mid\mid C_{\max}$ と LPT：補題 2

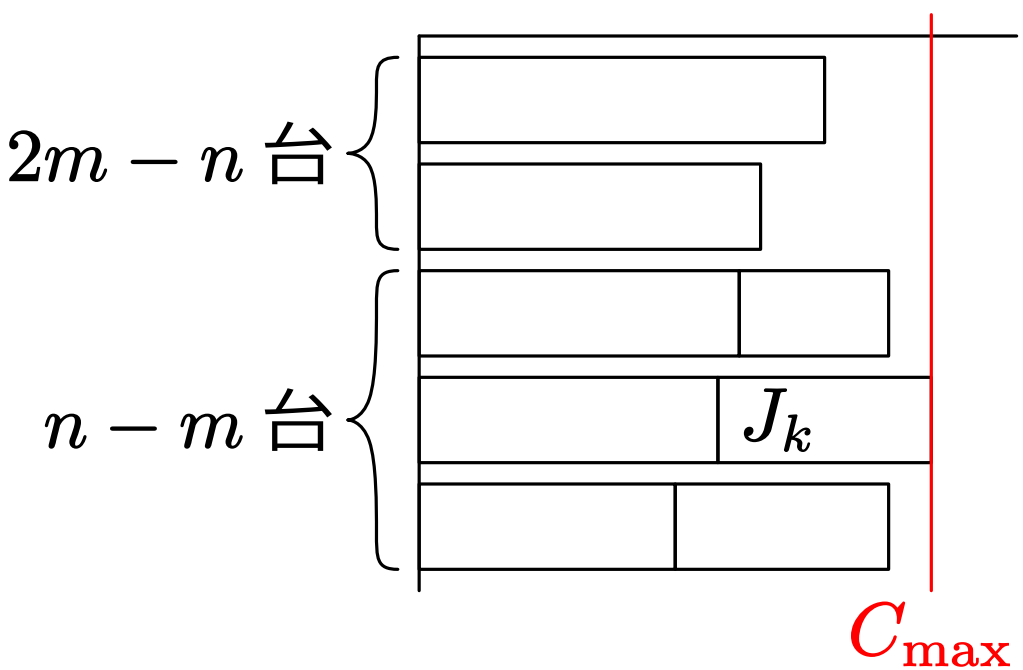
- LPT 順を $J_1, J_2, \ldots, J_n$ とする $(p_1 \geq p_2 \geq \cdots \geq p_n)$

補題 2

$p_n > C^*_{\max}/3 \Rightarrow C_{\max} = C^*_{\max}$

この仮定の下，最適解で各機械が処理するジョブ数 ≤ 2

- このとき，LPT が最適解を与える □



$P \mid\mid C_{\max}$ と LPT：補題 2

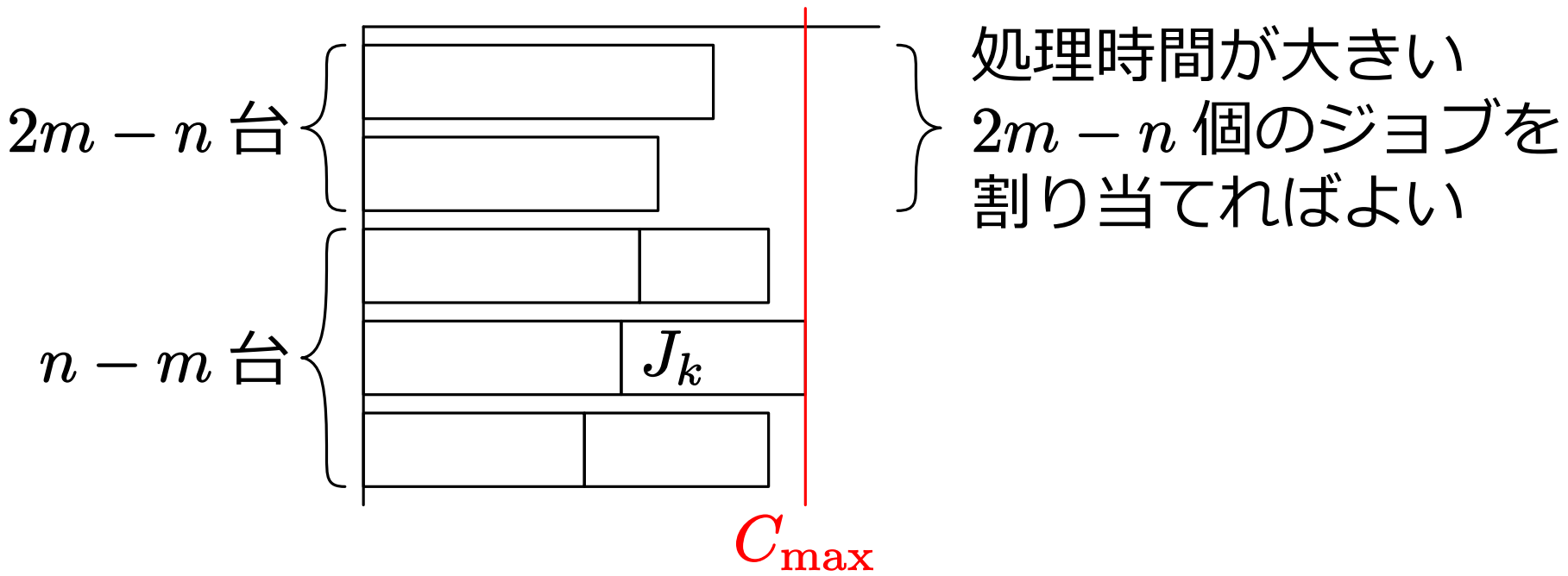
- LPT 順を $J_1, J_2, \ldots, J_n$ とする $(p_1 \geq p_2 \geq \cdots \geq p_n)$

補題 2

$p_n > C^*_{\max}/3 \Rightarrow C_{\max} = C^*_{\max}$

この仮定の下，最適解で各機械が処理するジョブ数 ≤ 2

- このとき，LPT が最適解を与える □



$P \mid\mid C_{\max}$ と LPT：補題 2

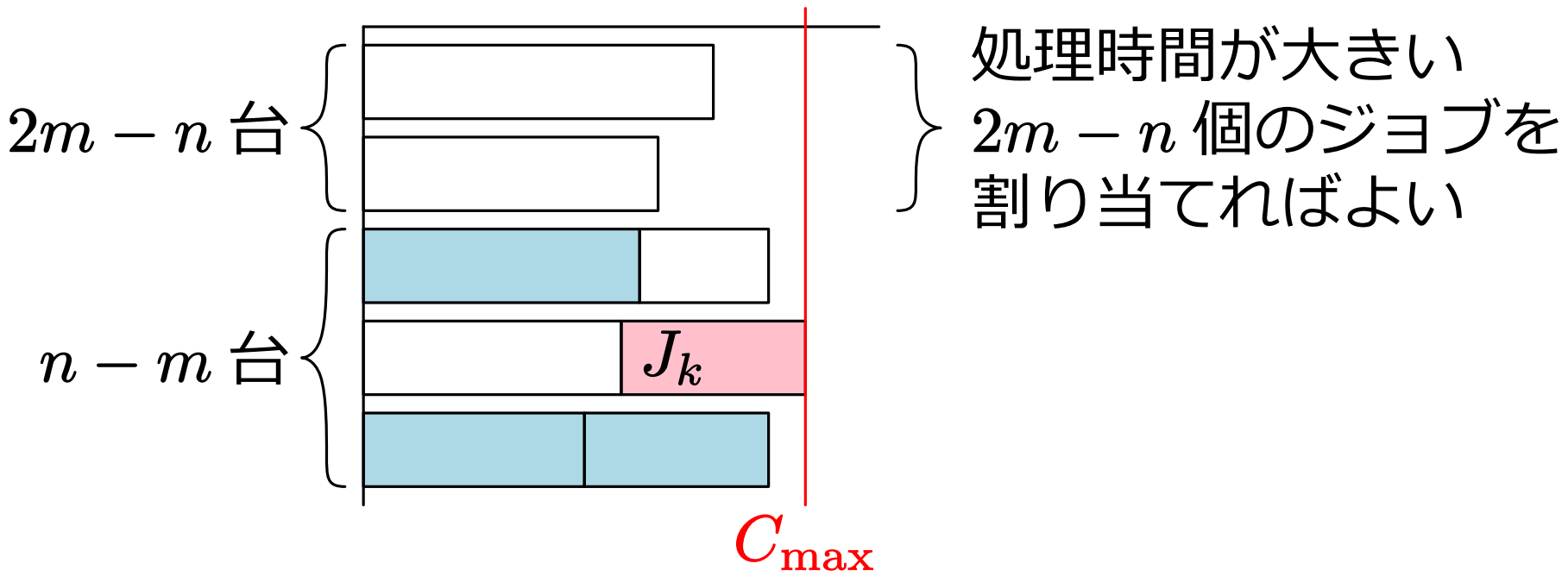
- LPT 順を $J_1, J_2, \ldots, J_n$ とする $(p_1 \geq p_2 \geq \cdots \geq p_n)$

補題 2

$p_n > C^*_{\max}/3 \Rightarrow C_{\max} = C^*_{\max}$

この仮定の下，最適解で各機械が処理するジョブ数 ≤ 2

- このとき，LPT が最適解を与える □



$P \mid\mid C_{\max}$ と LPT：補題 2

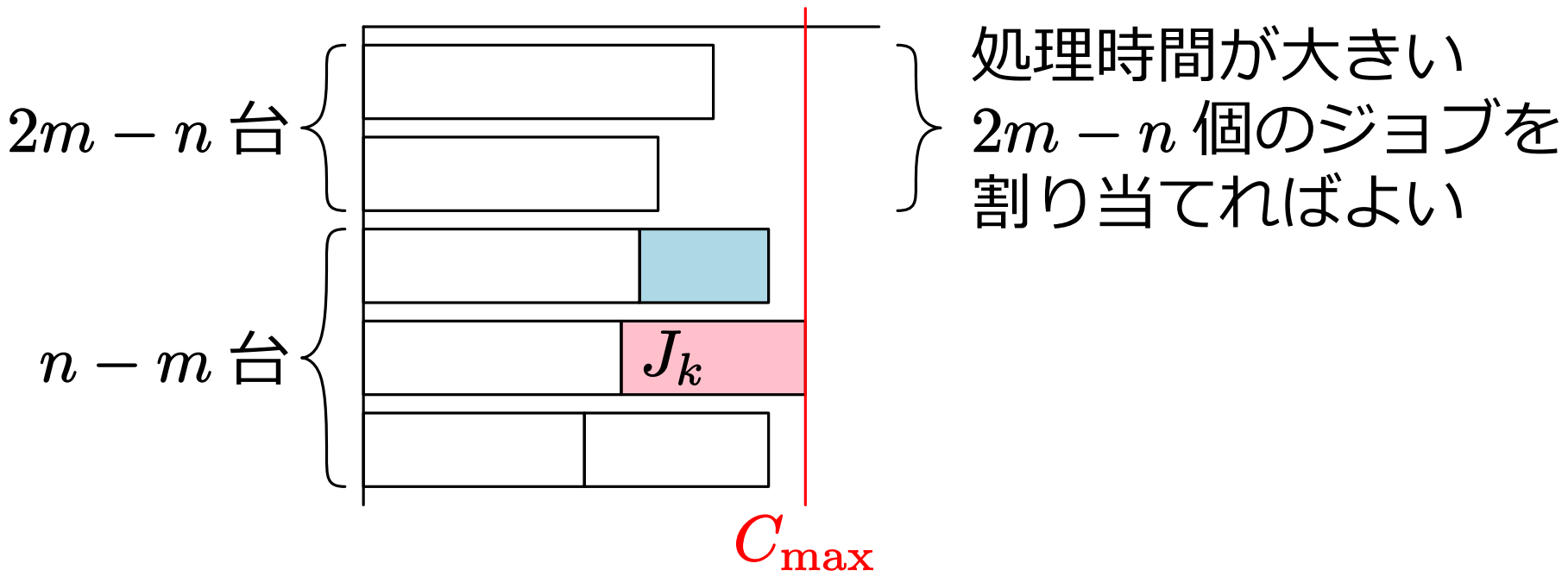
- LPT 順を $J_1, J_2, \ldots, J_n$ とする $(p_1 \geq p_2 \geq \cdots \geq p_n)$

補題 2

$p_n > C^*_{\max}/3 \Rightarrow C_{\max} = C^*_{\max}$

この仮定の下，最適解で各機械が処理するジョブ数 ≤ 2

- このとき，LPT が最適解を与える □



$P \mid\mid C_{\max}$ と LPT：結論

- LPT 順を $J_1, J_2, \ldots, J_n$ とする $(p_1 \geq p_2 \geq \cdots \geq p_n)$
- $C^*_{\max}$ は次を満たす　(任意のリストのときと同じ)
  - $C^*_{\max} \geq p_j$ $(\forall\, j \in [n])$
  - $C^*_{\max} \geq \dfrac{1}{m} \displaystyle\sum_{j=1}^{n} p_j$

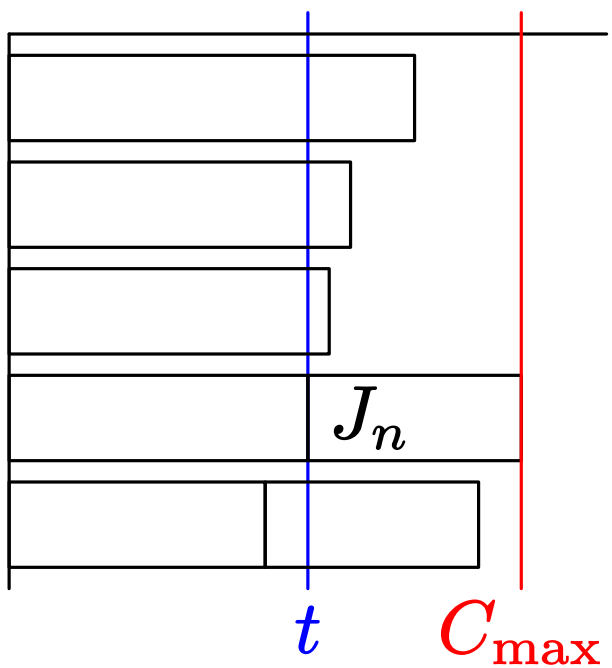
$P \mid\mid C_{\max}$ と LPT：結論

- LPT 順を $J_1, J_2, \ldots, J_n$ とする $(p_1 \geq p_2 \geq \cdots \geq p_n)$
- $C^*_{\max}$ は次を満たす　(任意のリストのときと同じ)
  - $C^*_{\max} \geq p_j \ (\forall\, j \in [n])$
  - $C^*_{\max} \geq \dfrac{1}{m} \displaystyle\sum_{j=1}^{n} p_j$
- 最後に処理が終わるジョブは J_n であるとしてよい

(∵ 補題 1)

- J_n の処理開始時刻を t とする
- このとき，次が成り立つ
  - $C_{\max} = t + p_n$
  - $t \leq \dfrac{1}{m}\left(\displaystyle\sum_{j=1}^{n} p_j - p_n\right)$

(任意のリストのときと同じ)



$P \mid\mid C_{\max}$ と LPT：結論 (続き)

- $p_n > C^*_{\max}/3$ のとき，$C_{\max} = C^*_{\max}$ (∵ 補題 2)
- $p_n \le C^*_{\max}/3$ のとき，

$$
\begin{aligned}
C_{\max} &\le \frac{1}{m}\left(\sum p_j - p_n\right) + p_n \\
&= \frac{1}{m}\sum p_j + \left(1 - \frac{1}{m}\right) p_n \\
&\le C^*_{\max} + \left(1 - \frac{1}{m}\right) \cdot \frac{1}{3} C^*_{\max} = \left(\frac{4}{3} - \frac{1}{3m}\right) C^*_{\max}
\end{aligned}
$$

$P \mid\mid C_{\max}$ と LPT：結論 (続き)

- $p_n > C^*_{\max}/3$ のとき，$C_{\max} = C^*_{\max}$ (∵ 補題 2)
- $p_n \leq C^*_{\max}/3$ のとき，

$$
\begin{aligned}
C_{\max} &\leq \frac{1}{m}\left(\sum p_j - p_n\right) + p_n \\
&= \frac{1}{m}\sum p_j + \left(1 - \frac{1}{m}\right) p_n \\
&\leq C^*_{\max} + \left(1 - \frac{1}{m}\right) \cdot \frac{1}{3} C^*_{\max} = \left(\frac{4}{3} - \frac{1}{3m}\right) C^*_{\max}
\end{aligned}
$$

- したがって，いずれの場合においても，

$$
C_{\max} \leq \left(\frac{4}{3} - \frac{1}{3m}\right) C^*_{\max}
$$

□

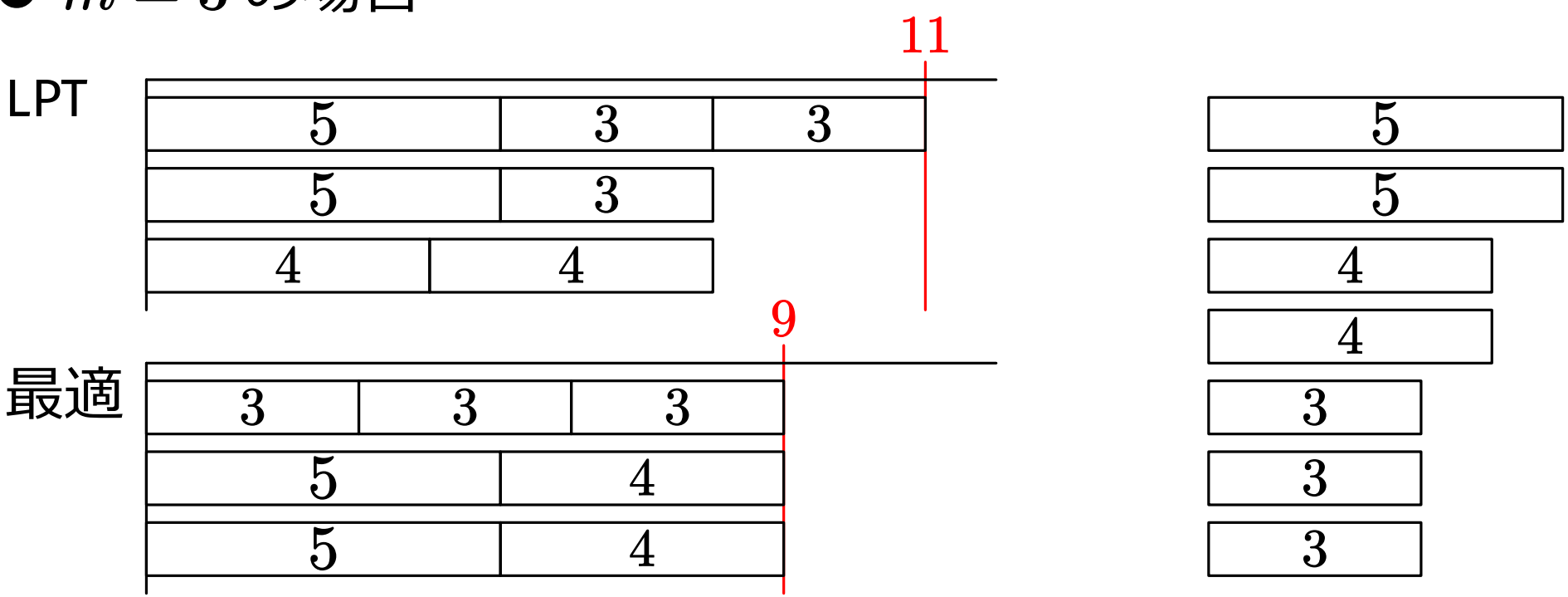
$P \mid\mid C_{\max}$：LPT (タイト)

性質：近似比がタイト

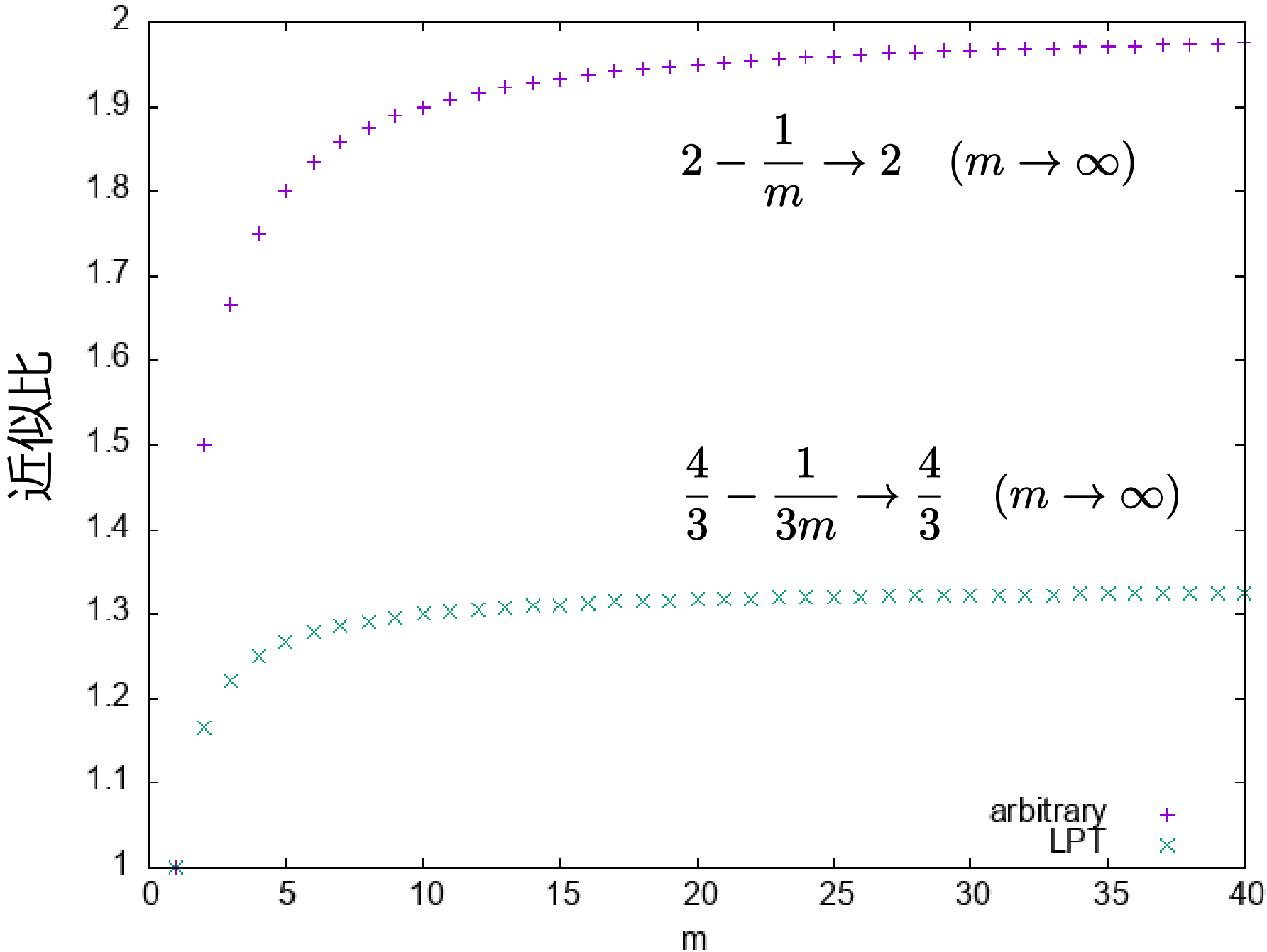
どの整数 $m \geq 1$ に対しても，
LPT に対するリスト・スケジューリングの
近似比は $\frac{4}{3} - \frac{1}{3m}$ より小さくない

つまり，$\frac{4}{3} - \frac{1}{3m}$ という近似比は タイト

- $m = 3$ の場合

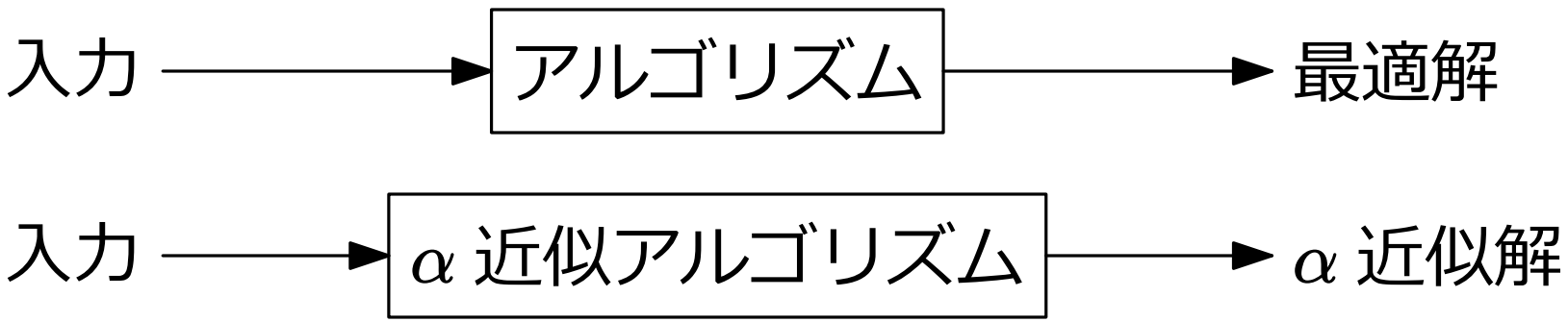


$P \mid\mid C_{\max}$：任意リストと LPT



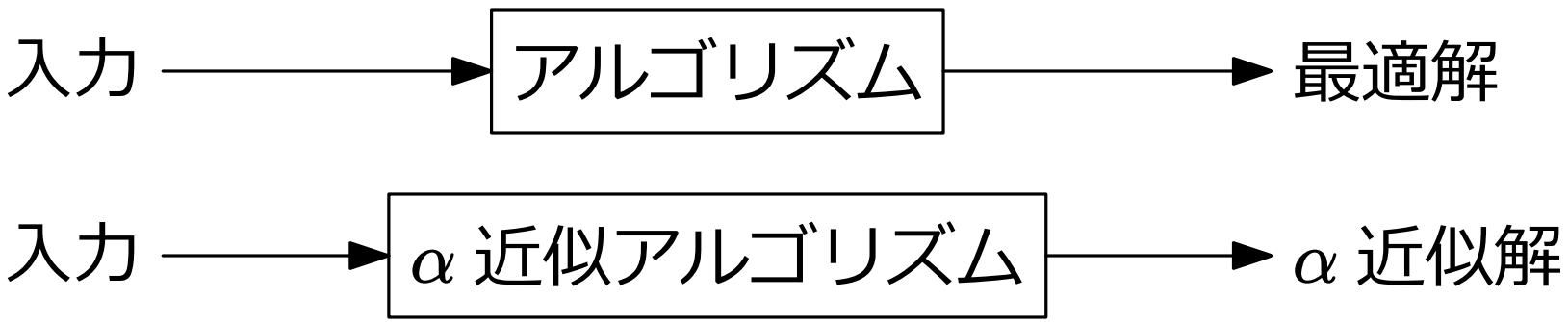
アルゴリズムの定義の反省

- アルゴリズムは「正しい出力」を行わなければならないはずだった
- Q：近似アルゴリズムの出力は「正しい」のか？



アルゴリズムの定義の反省

- アルゴリズムは「正しい出力」を行わなければならないはずだった
- Q：近似アルゴリズムの出力は「正しい」のか？



考え方 のひとつ

- 問題における出力に対する要求を変える
- 例えば，「α 近似解」を「正しい出力」とすれば，α 近似アルゴリズムは正しいアルゴリズムである

注意：近似アルゴリズムは元の問題を解いていない

本日の内容

1. $P \mid\mid C_{\max}$ に対するリスト・スケジューリング
2. $P \mid\mid \sum C_j$ **に対するリスト・スケジューリング**
3. $P \mid\mid \sum w_j C_j$ に対するリスト・スケジューリング

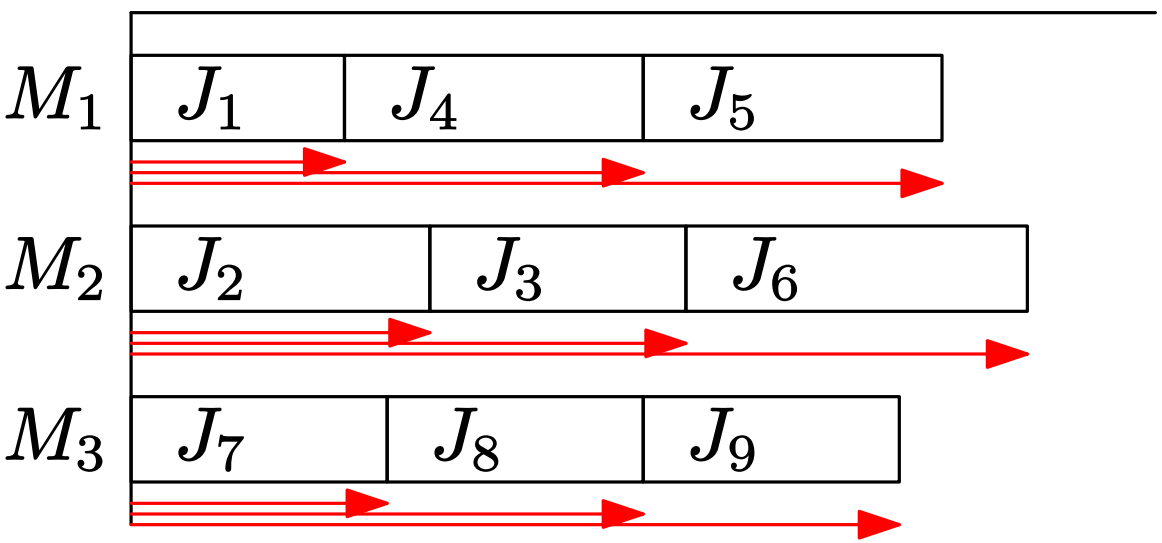
- R. W. Conway, W. L. Maxwell, L. W. Miller, *Theory of Scheduling*, Addison-Wesley, 1967.

機械の環境

- 一様並列機械，機械数 $= m$ (入力の一部)
  - $p_j =$ ジョブ J_j の処理時間

最適化する目的

- 総完了時刻の最小化



復習：問題 $P \mid\mid \sum C_j$ は $O(m^3 n^3)$ 時間で解ける

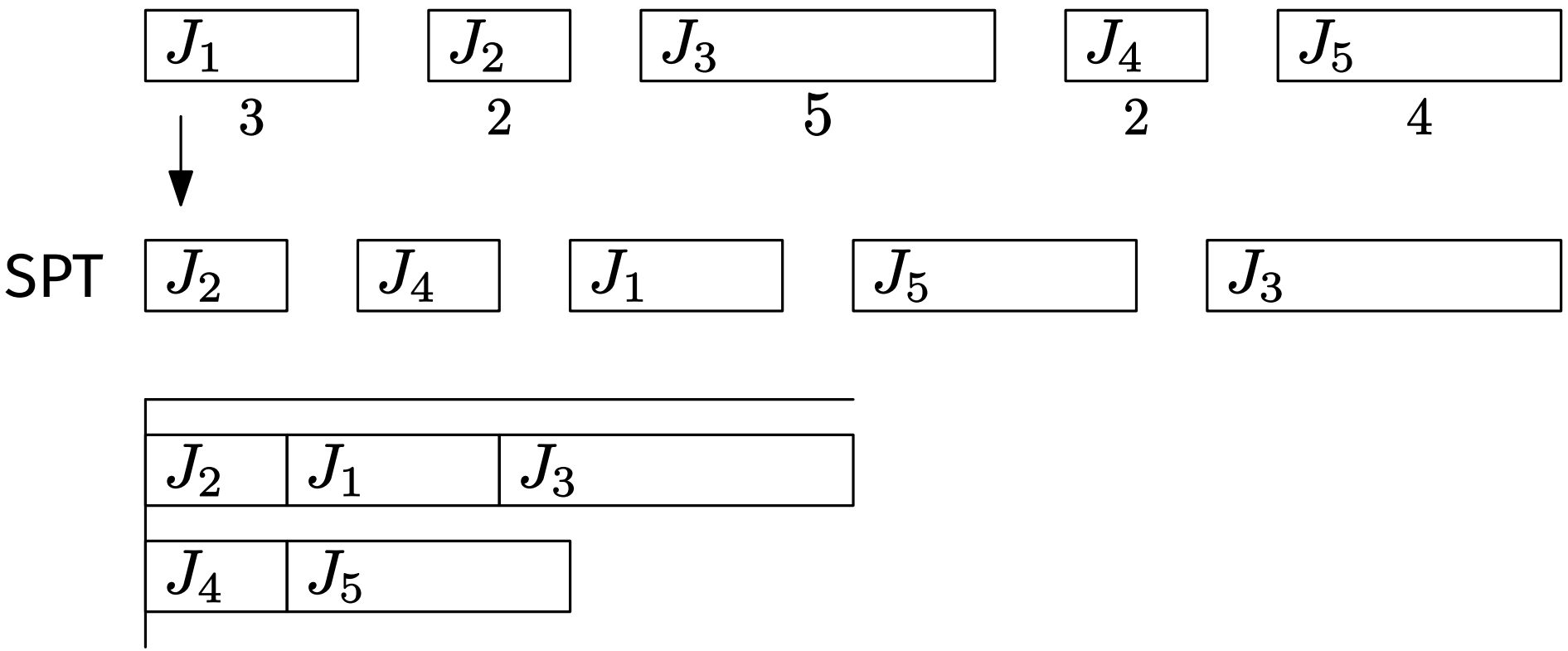
$P \| \sum C_j$ をより速く解く

定理

$P \| \sum C_j$ は $O(m + n \log n)$ 時間で解ける

アルゴリズム：SPT に対するリスト・スケジューリング

SPT = Shortest Processing Time (最短処理時間優先規則)



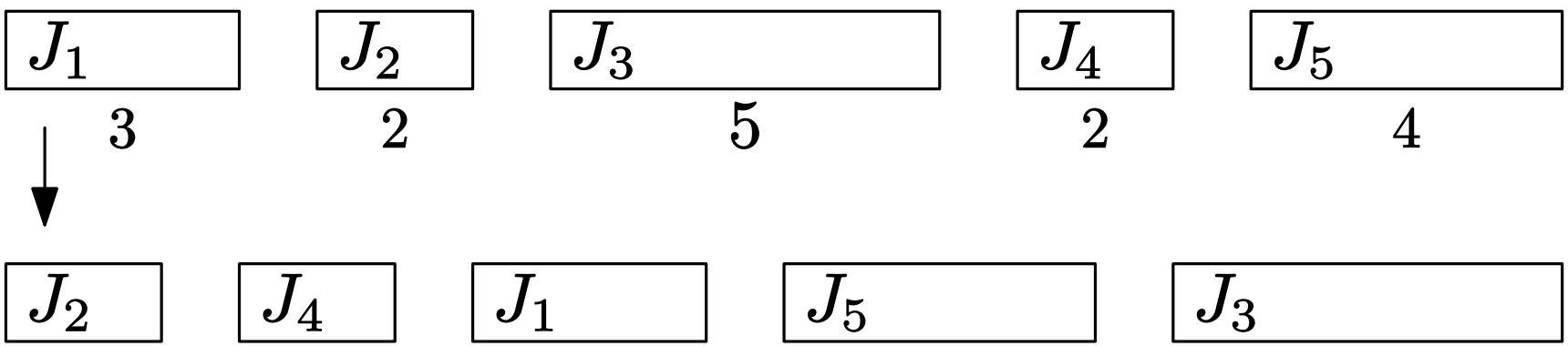
$P \mid\mid \sum C_j$ をより速く解く

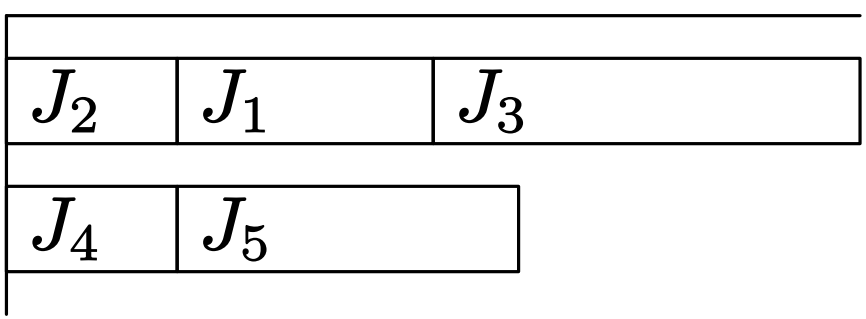
定理

$P \mid\mid \sum C_j$ は $O(m + n \log n)$ 時間で解ける

アルゴリズム：SPT に対するリスト・スケジューリング

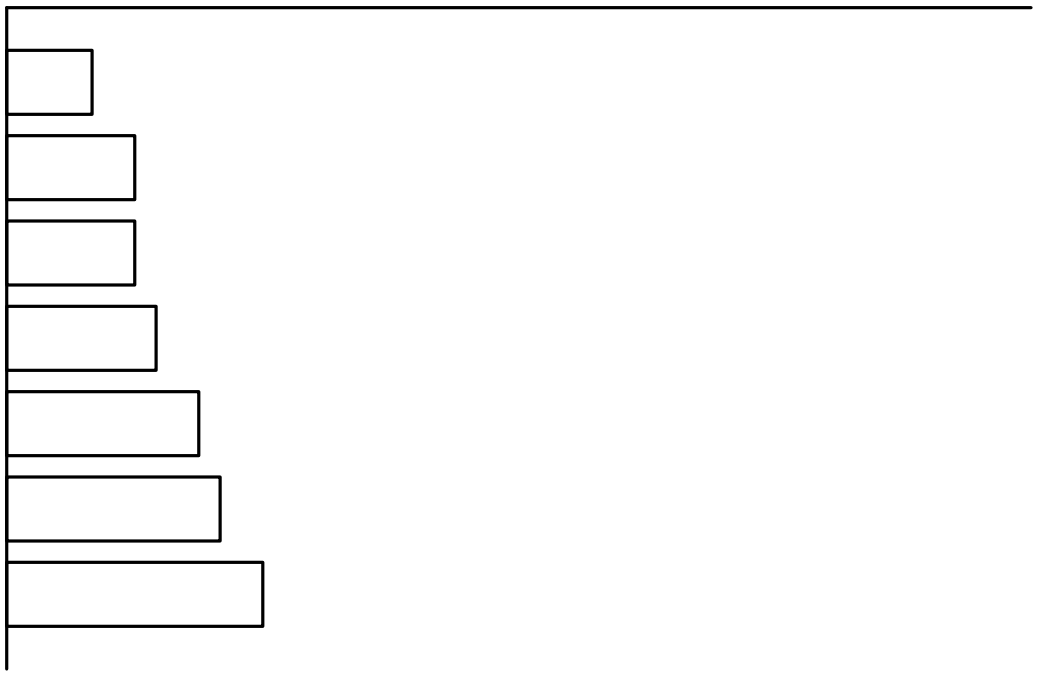
SPT = Shortest Processing Time (最短処理時間優先規則)



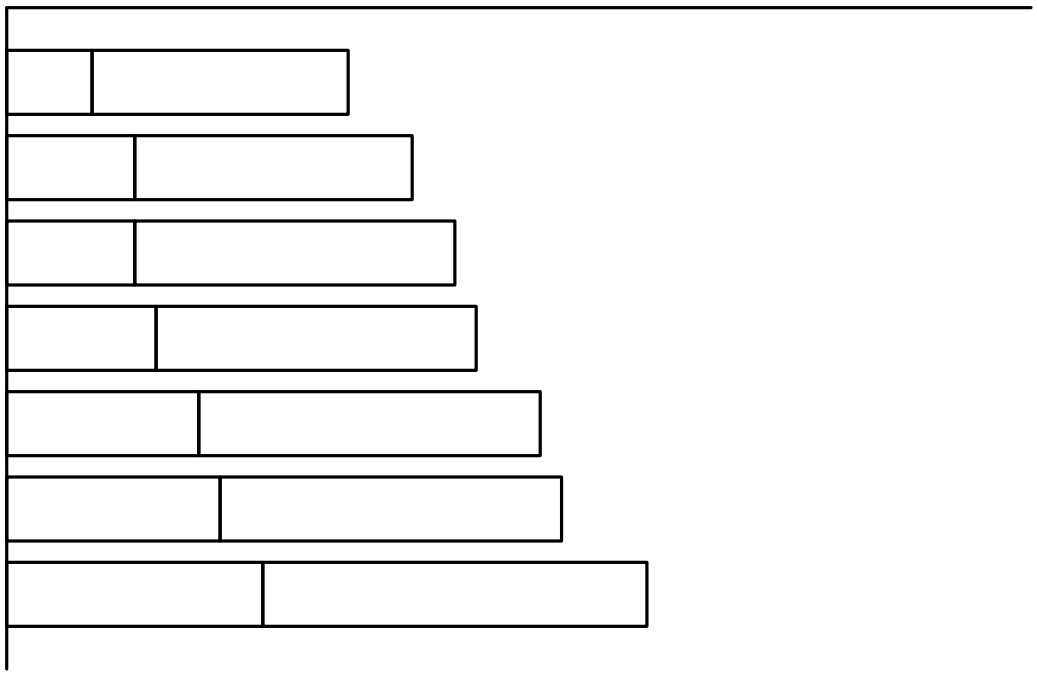


復習：
$1 \mid\mid \sum C_j$ は SPT で解ける

$P \mid\mid \sum C_j$ とSPT：証明 (の気分)

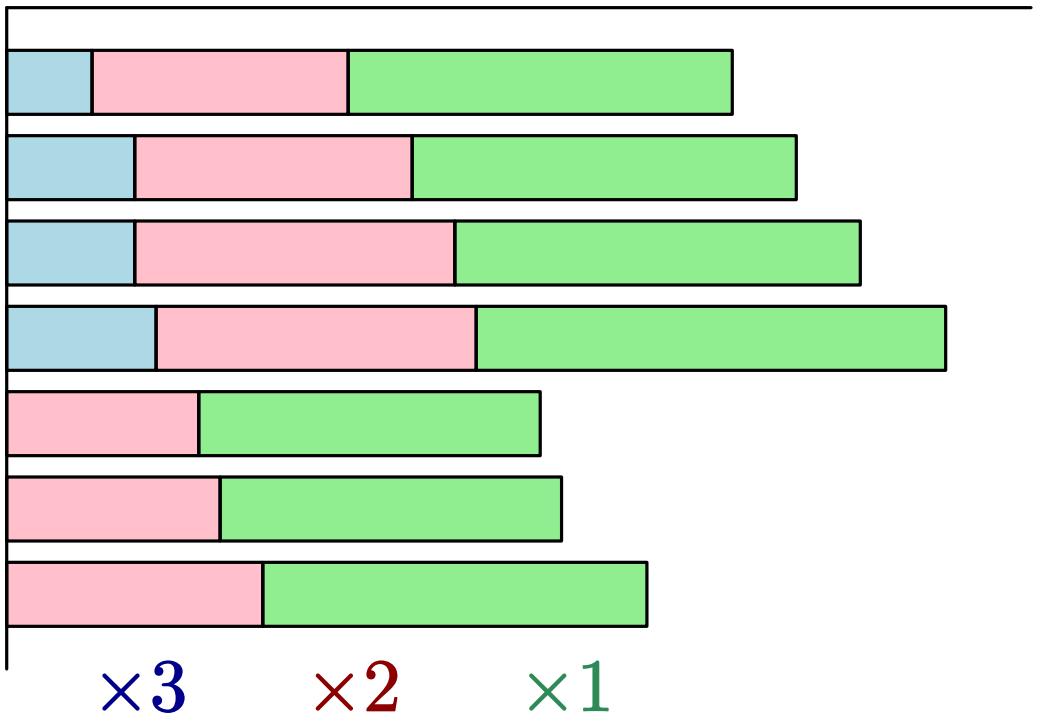


$P \mid\mid \sum C_j$ とSPT：証明 (の気分)



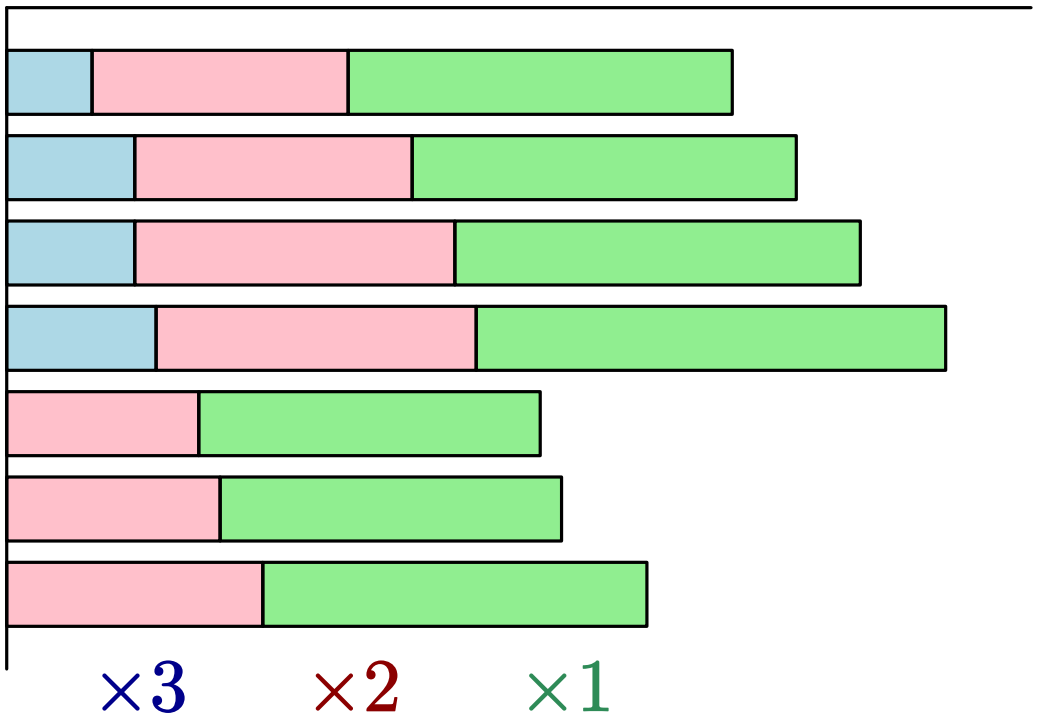
$P \mid\mid \sum C_j$ とSPT：証明 (の気分)

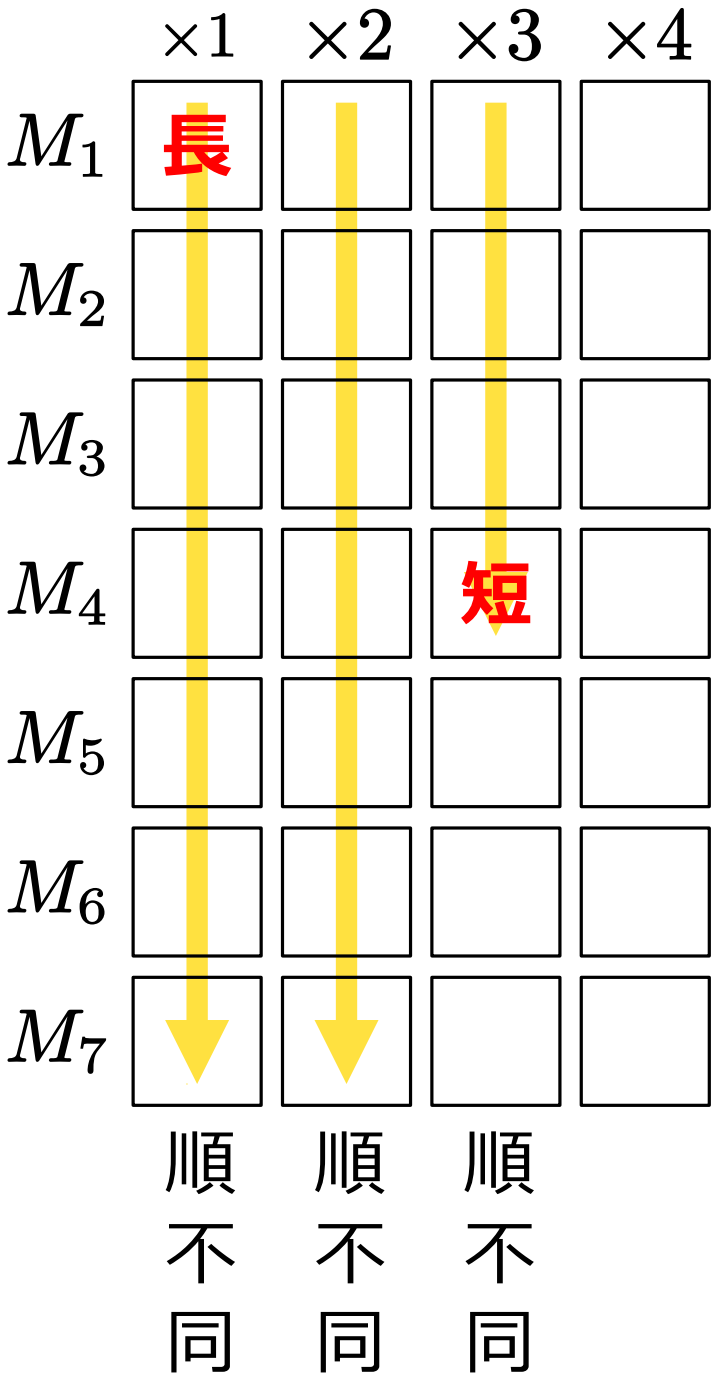
$P \mid\mid \sum C_j$ とSPT：証明 (の気分)



	×1	×2	×3	×4
M_1				
M_2				
M_3				
M_4				
M_5				
M_6				
M_7				

$P \| \sum C_j$ とSPT：証明 (の気分)





本日の内容

1. $P \,||\, C_{\max}$ に対するリスト・スケジューリング
2. $P \,||\, \sum C_j$ に対するリスト・スケジューリング
3. $P \,||\, \sum w_j C_j$ **に対するリスト・スケジューリング**

- T. Kawaguchi, S. Kyan, Worst case bound of an LRF schedule for the mean weighted flow time problem. *SIAM Journal of Computing* (1986) pp. 1119–1129.
- U. Schwiegelshohn, An alternative proof of the Kawaguchi-Kyan bound for the Largest-Ratio-First rule. *Operations Research Letters* 39 (2011) pp. 255–259.

T. Kawaguchi = 川口 剛 (かわぐち つよし)

S. Kyan = 喜屋武 盛基 (きゃん せいき)

$P \mid\mid \sum w_j C_j$

機械の環境

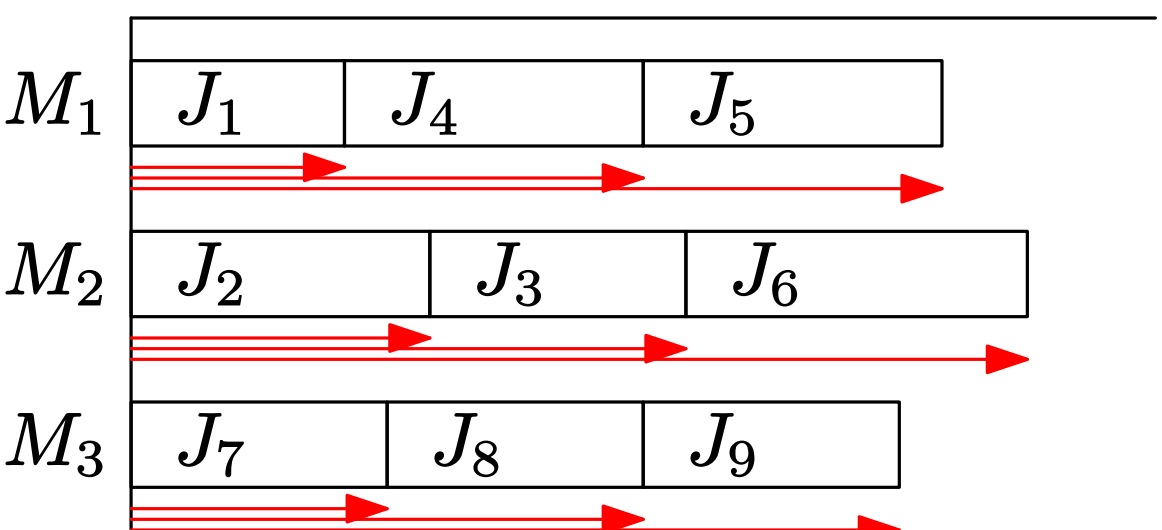
- 一様並列機械，機械数 $= m$ (入力の一部)
  - $p_j =$ ジョブ J_j の処理時間

ジョブの特性

- $w_j =$ ジョブ J_j の重み

最適化する目的

- 重み付き総完了時刻の最小化



事実：問題 $P \mid\mid \sum w_j C_j$ は強 NP 困難 (Lawler?)

$P \mid\mid \sum w_j C_j$ と WSPT

定理 (Kawaguchi, Kyan '86)

$P \mid\mid \sum w_j C_j$ に対して
WSPT に対するリスト・スケジューリングは
$\frac{1}{2}(1+\sqrt{2})$ 近似アルゴリズムである $\frac{1}{2}(1+\sqrt{2}) < 1.20711$

WSPT = Weighted Shortest Processing Time

$$\frac{p_1}{w_1} \leq \frac{p_2}{w_2} \leq \cdots \leq \frac{p_n}{w_n}$$

証明を完全に紹介するのは大変なので,
証明の概要だけを紹介する (付録もない)

紹介するものは Schwiegelshohn '11 による

補題 A

$w_j = p_j$ のときに証明すれば十分

以下，$w_j = p_j$ のときのみを考える

$P \mid\mid \sum w_j C_j$ と WSPT：証明 (1)

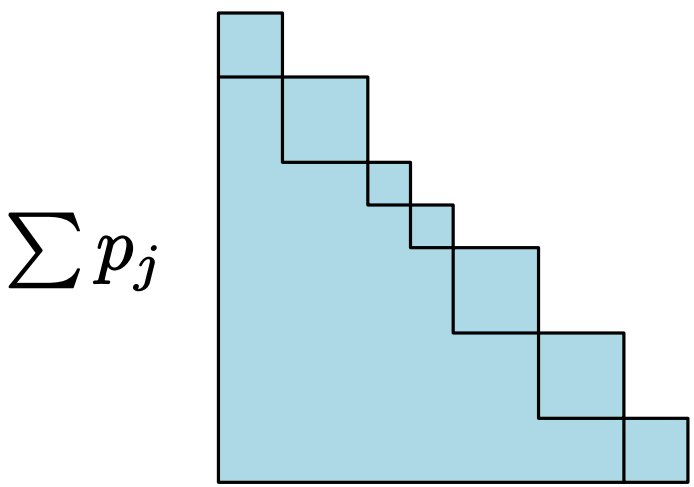
補題 A

$w_j = p_j$ のときに証明すれば十分

以下，$w_j = p_j$ のときのみを考える

補題 B (Queyranne '93)

一機械において，$\sum p_j C_j = \frac{1}{2}\left(\sum p_j^2 + \left(\sum p_j\right)^2\right)$



$P \mid\mid \sum w_j C_j$ と WSPT：証明 (1)

補題 A

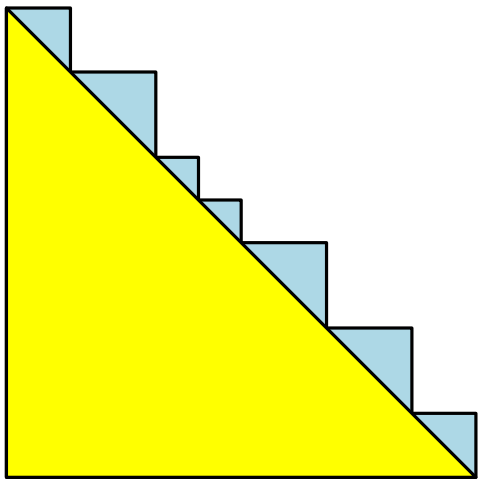
$w_j = p_j$ のときに証明すれば十分

以下，$w_j = p_j$ のときのみを考える

補題 B (Queyranne '93)

一機械において，$\sum p_j C_j = \frac{1}{2}\left(\sum p_j^2 + \left(\sum p_j\right)^2\right)$

$\sum p_j$



$P \mid\mid \sum w_j C_j$ と WSPT：証明 (2)

補題 C

次の入力を考えれば十分

- すべての j について，$w_j = p_j$
- $< m$ 個のジョブの処理時間が同じ　　(大きなジョブ)
- 他のジョブの処理時間は微小で同じ　　(小さなジョブ)

$P \mid\mid \sum w_j C_j$ と WSPT：証明 (2)

補題 C

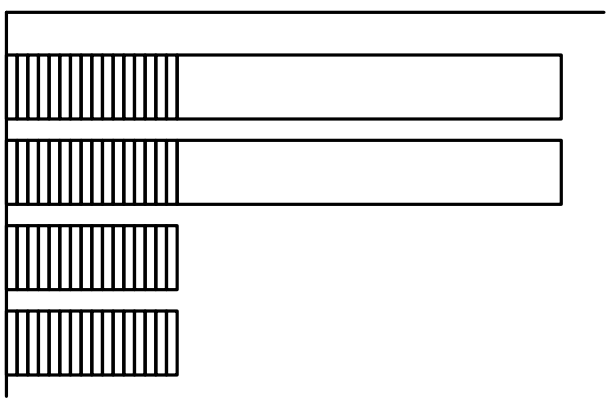
次の入力を考えれば十分

- すべての j について，$w_j = p_j$
- $< m$ 個のジョブの処理時間が同じ　　(大きなジョブ)
- 他のジョブの処理時間は微小で同じ　　(小さなジョブ)

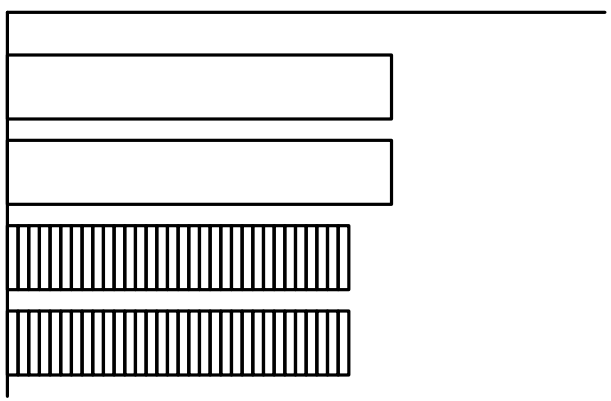
次のパラメータを用意する

- $u =$ 大きなジョブの数
- $p =$ 大きなジョブの処理時間
- $\varepsilon =$ 小さなジョブの処理時間

WSPT



最適



$P \mid\mid \sum w_j C_j$ と WSPT：証明 (2)

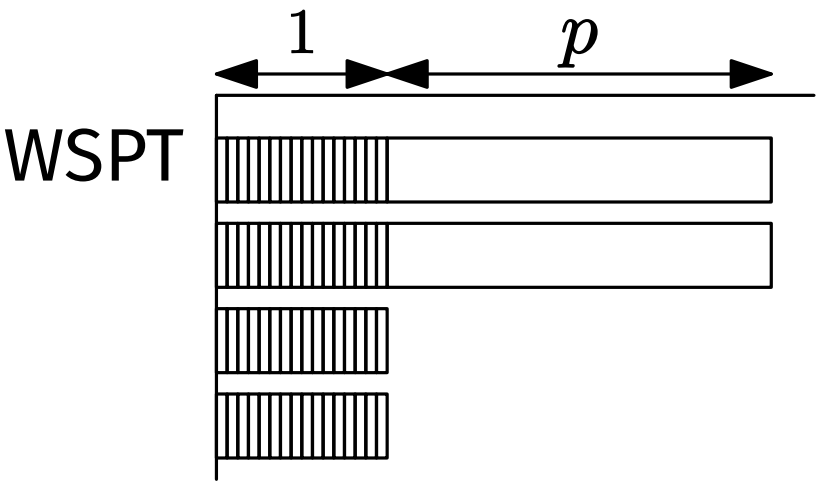
補題 C

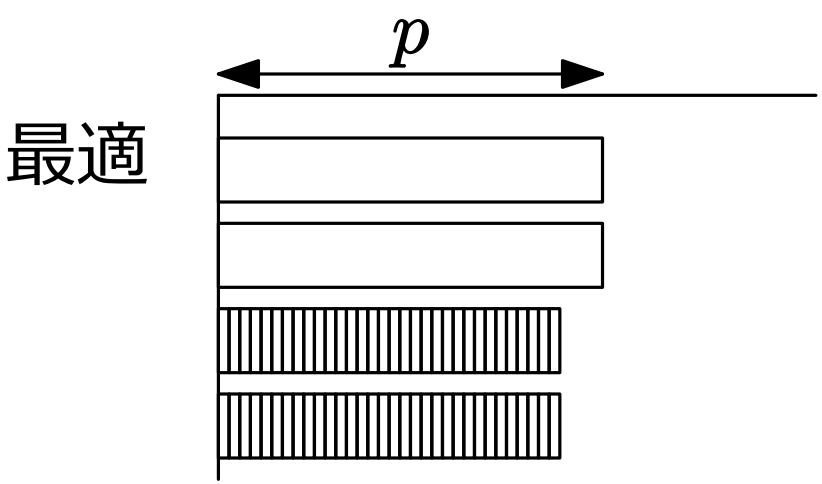
次の入力を考えれば十分

- すべての j について，$w_j = p_j$
- $< m$ 個のジョブの処理時間が同じ　　(大きなジョブ)
- 他のジョブの処理時間は微小で同じ　　(小さなジョブ)

次のパラメータを用意する

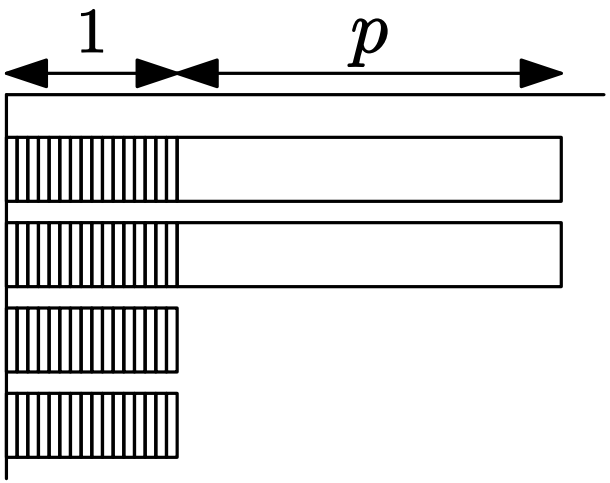
- $u =$ 大きなジョブの数
- $p =$ 大きなジョブの処理時間
- $\varepsilon =$ 小さなジョブの処理時間



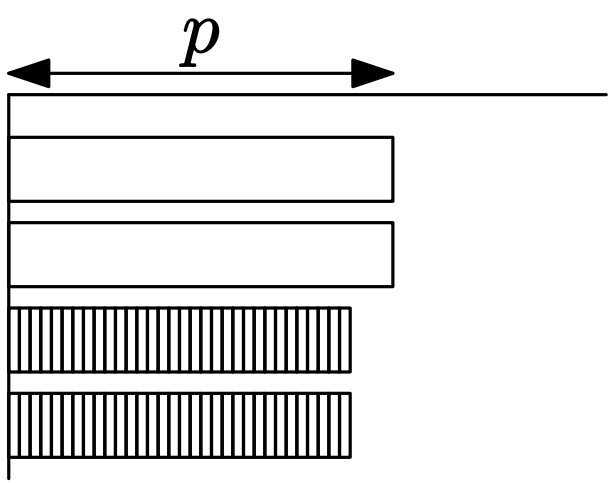


$P \mid\mid \sum w_j C_j$ と WSPT：証明 (3)

WSPT

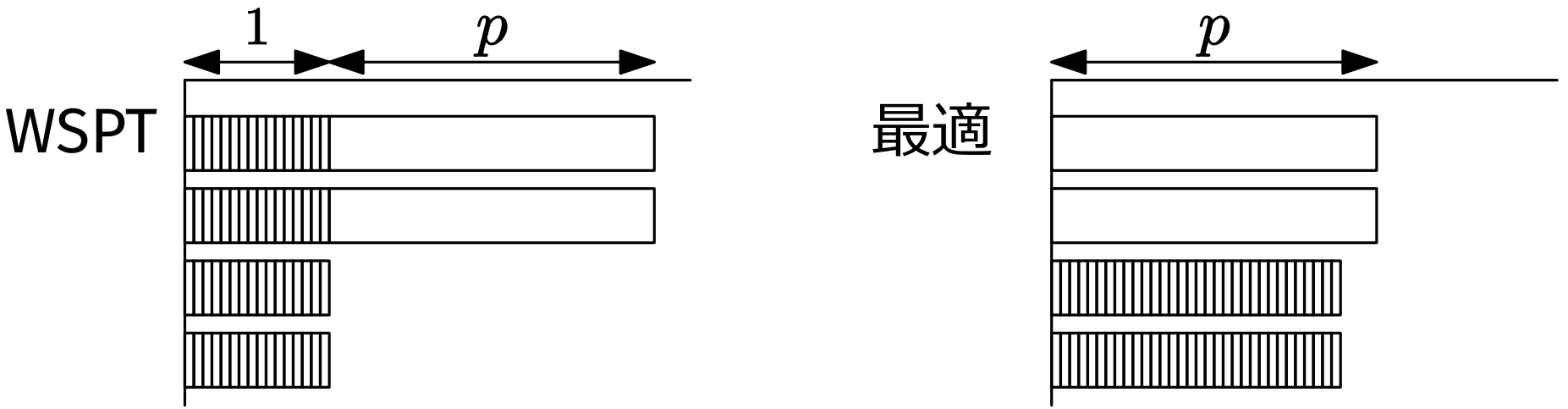


最適



$$\text{近似比} = \frac{\frac{u}{2}(p^2 + (1+p)^2) + \frac{m-u}{2} + O(\varepsilon^2)}{up^2 + \frac{m-u}{2}\left(\frac{m}{m-u}\right)^2 + O(\varepsilon^2)}$$

$P \mid\mid \sum w_j C_j$ と WSPT：証明 (3)

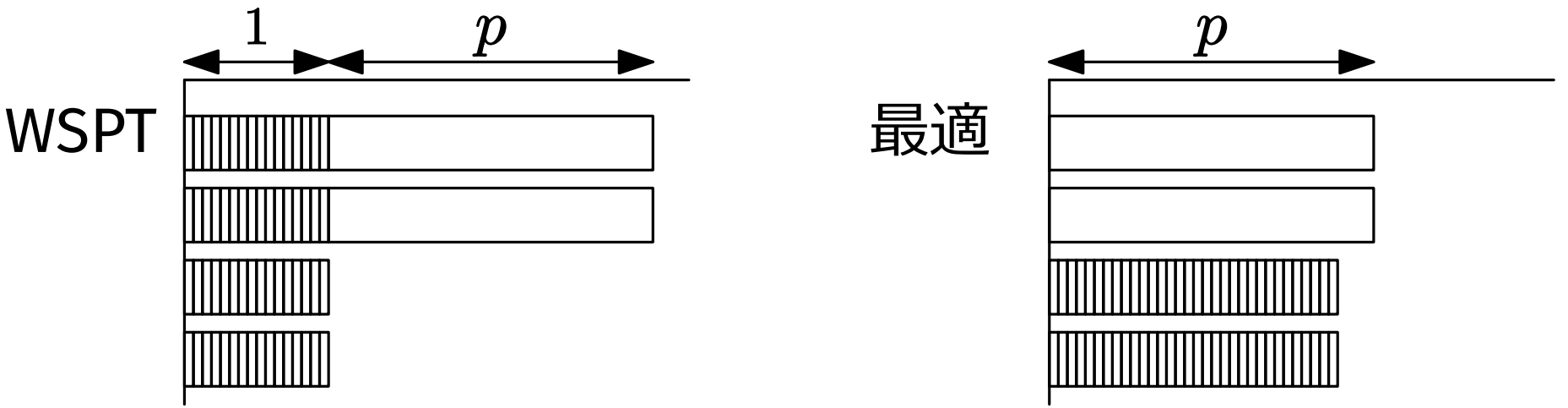


$$\text{近似比} = \frac{\frac{u}{2}(p^2+(1+p)^2)+\frac{m-u}{2}+O(\varepsilon^2)}{up^2+\frac{m-u}{2}\left(\frac{m}{m-u}\right)^2+O(\varepsilon^2)}$$

$$= \frac{x(p^2+(1+p)^2)+(1-x)+O(\varepsilon^2)}{2xp^2+\frac{1}{1-x}+O(\varepsilon^2)} \qquad \left(x=\frac{u}{m}\right)$$

$$\to \frac{x(p^2+(1+p)^2)+(1-x)}{2xp^2+\frac{1}{1-x}} \qquad (\varepsilon \to 0)$$

$P \mid\mid \sum w_j C_j$ とWSPT：証明 (3)



$$\text{近似比} = \frac{\frac{u}{2}(p^2+(1+p)^2)+\frac{m-u}{2}+O(\varepsilon^2)}{up^2+\frac{m-u}{2}\left(\frac{m}{m-u}\right)^2+O(\varepsilon^2)} \qquad p \geq \frac{m}{m-u}$$

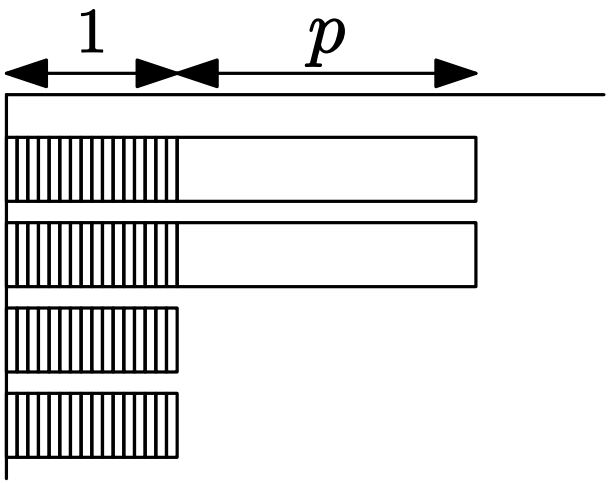
$$= \frac{x(p^2+(1+p)^2)+(1-x)+O(\varepsilon^2)}{2xp^2+\frac{1}{1-x}+O(\varepsilon^2)} \qquad \left(x=\frac{u}{m}\right)$$

$$\to \frac{x(p^2+(1+p)^2)+(1-x)}{2xp^2+\frac{1}{1-x}} \qquad (\varepsilon \to 0)$$

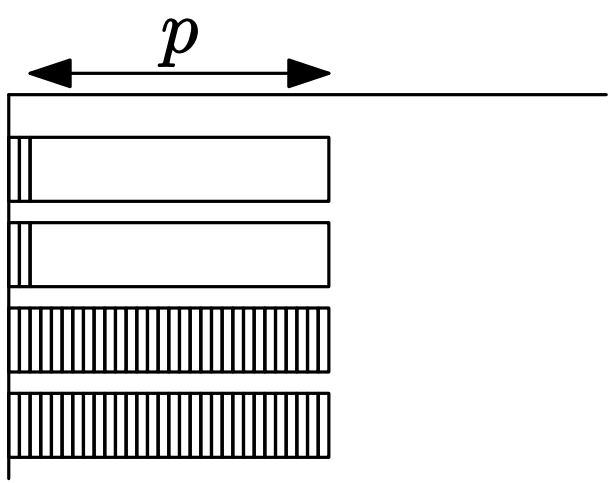
$$0 < x < 1, p \geq \frac{1}{1-x}$$

$P \mid\mid \sum w_j C_j$ と WSPT：証明 (4)

WSPT



最適

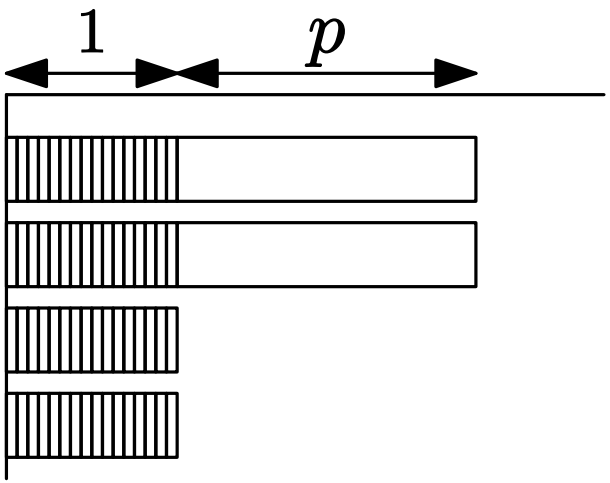


$$\text{近似比} = \frac{\frac{u}{2}(p^2 + (1+p)^2) + \frac{m-u}{2} + O(\varepsilon^2)}{\frac{u}{2}p^2 + \frac{m}{2}\left(\frac{up+m}{m}\right)^2 + O(\varepsilon^2)}$$

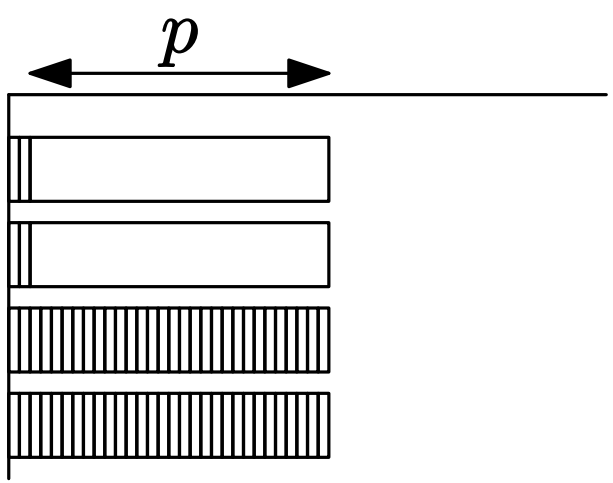
$$p \leq \frac{m}{m-u}$$

$P \mid\mid \sum w_j C_j$ と WSPT：証明 (4)

WSPT



最適

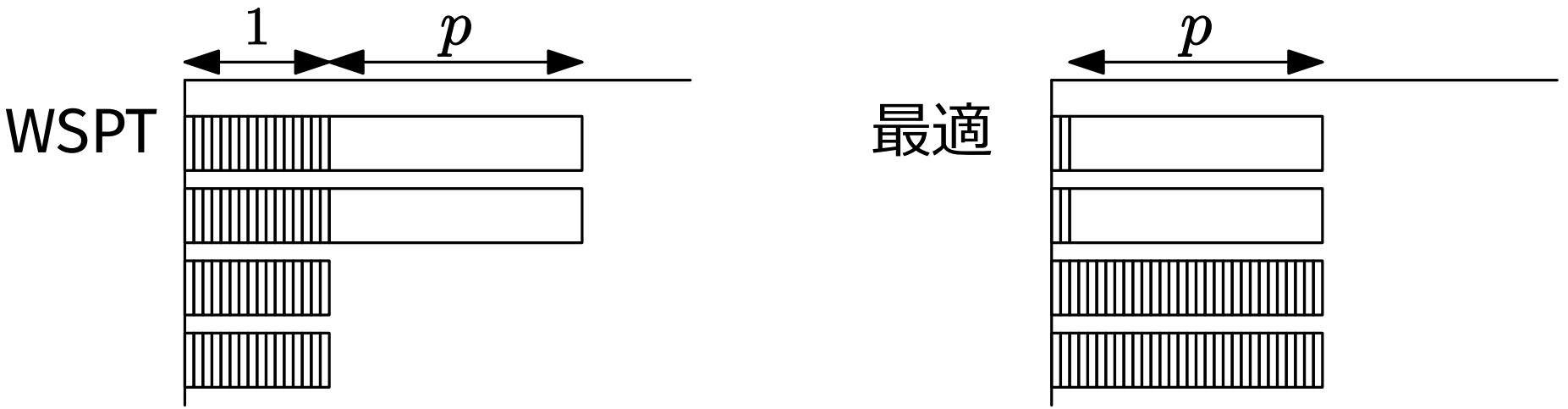


$$\text{近似比} = \frac{\frac{u}{2}(p^2+(1+p)^2)+\frac{m-u}{2}+O(\varepsilon^2)}{\frac{u}{2}p^2+\frac{m}{2}\left(\frac{up+m}{m}\right)^2+O(\varepsilon^2)} \qquad p \le \frac{m}{m-u}$$

$$= \frac{x(p^2+(1+p)^2)+(1-x)+O(\varepsilon^2)}{xp^2+(xp+1)^2+O(\varepsilon^2)} \qquad \left(x=\frac{u}{m}\right)$$

$$\to \frac{x(p^2+(1+p)^2)+(1-x)}{xp^2+(xp+1)^2} \qquad (\varepsilon \to 0)$$

$P \mid\mid \sum w_j C_j$ とWSPT：証明 (4)



$$\text{近似比} = \frac{\frac{u}{2}(p^2+(1+p)^2)+\frac{m-u}{2}+O(\varepsilon^2)}{\frac{u}{2}p^2+\frac{m}{2}\left(\frac{up+m}{m}\right)^2+O(\varepsilon^2)} \qquad p \le \frac{m}{m-u}$$

$$= \frac{x(p^2+(1+p)^2)+(1-x)+O(\varepsilon^2)}{xp^2+(xp+1)^2+O(\varepsilon^2)} \qquad (x=\frac{u}{m})$$

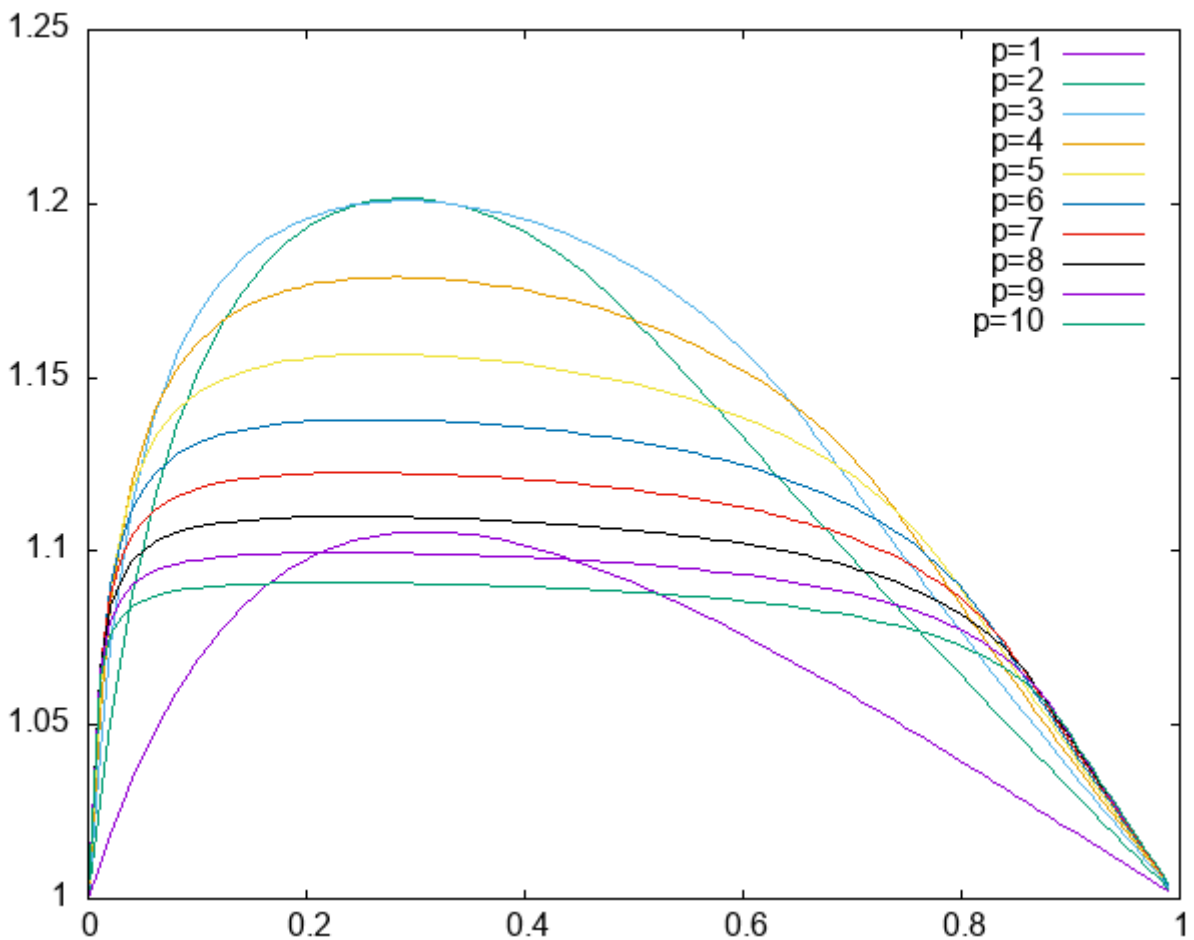
$$\to \frac{x(p^2+(1+p)^2)+(1-x)}{xp^2+(xp+1)^2} \qquad (\varepsilon \to 0)$$

$$0 < x < 1, p \le \frac{1}{1-x}$$

$P \mid\mid \sum w_j C_j$ とWSPT：証明 (5)

$$\text{近似比} = \begin{cases} \dfrac{x(p^2+(1+p)^2)+(1-x)}{xp^2+(xp+1)^2} & (p \le \frac{1}{1-x}) \\ \dfrac{x(p^2+(1+p)^2)+(1-x)}{2xp^2+\frac{1}{1-x}} & (p \ge \frac{1}{1-x}) \end{cases}$$

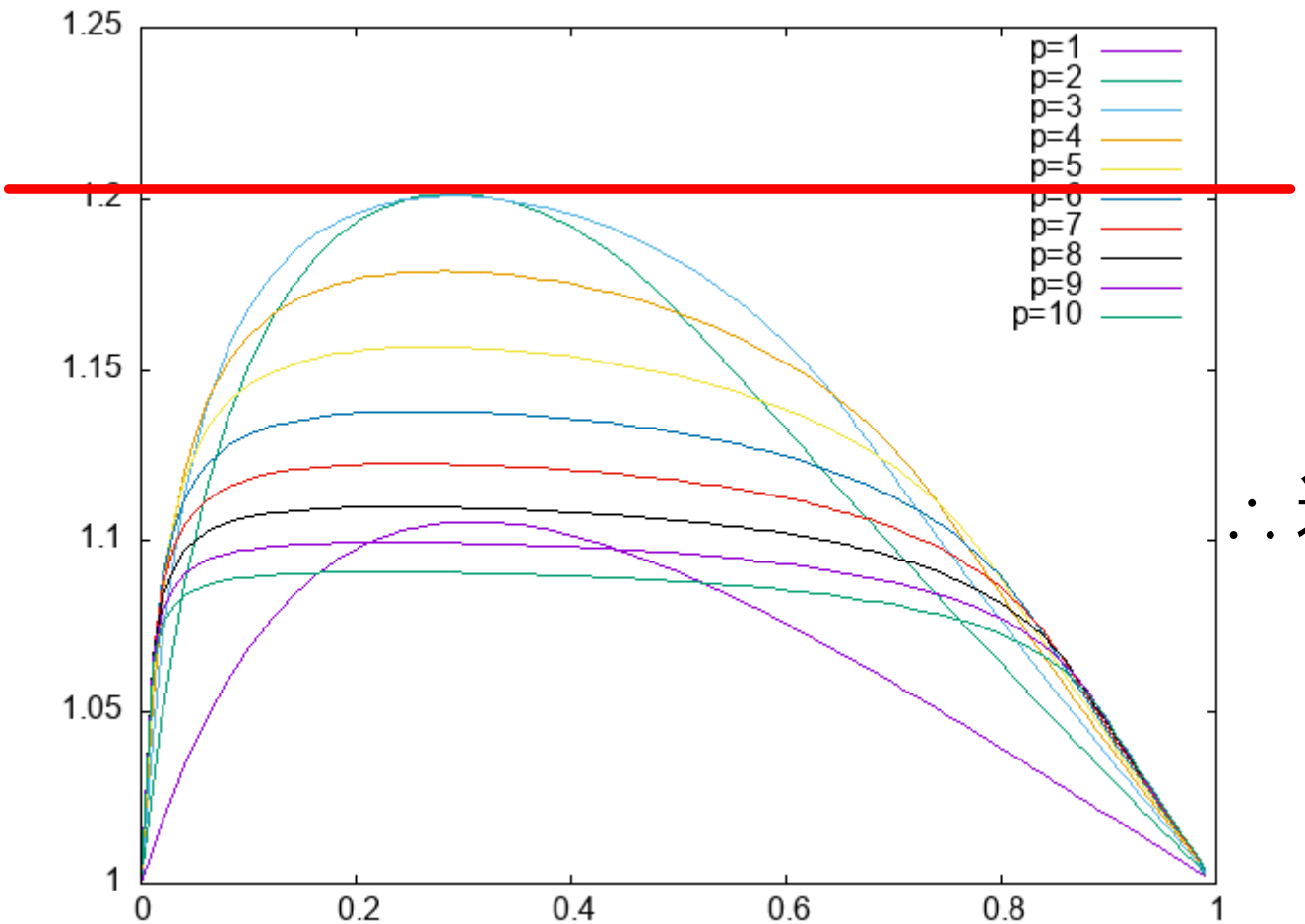
$$(0 < x < 1)$$



$P \mid\mid \sum w_j C_j$ と WSPT：証明 (5)

$$\text{近似比} = \begin{cases} \dfrac{x(p^2+(1+p)^2)+(1-x)}{xp^2+(xp+1)^2} & (p \le \frac{1}{1-x}) \\ \dfrac{x(p^2+(1+p)^2)+(1-x)}{2xp^2+\frac{1}{1-x}} & (p \ge \frac{1}{1-x}) \end{cases}$$

$(0 < x < 1)$



$p = 1+\sqrt{2}$ のとき

最大値 $\frac{1}{2}(1+\sqrt{2})$

$\therefore$ 近似比 $\le \frac{1}{2}(1+\sqrt{2})$

リスト・スケジューリングの近似比

問題	リスト	近似比	
$1 \mid\mid C_{\max}$	任意	1	
$1 \mid\mid \sum C_j$	SPT	1	(Smith '56)
$1 \mid\mid L_{\max}$	EDD	1	(Jackson '55)
$1 \mid\mid \sum w_j C_j$	WSPT	1	(Smith '56)
$P \mid\mid C_{\max}$	任意	$2 - \frac{1}{m}$	(Graham '66)
	LPT	$\frac{4}{3} - \frac{1}{3m}$	(Graham '69)
$P \mid\mid \sum C_j$	SPT	1	(Conway, Maxwell, Miller '67)
$P \mid\mid \sum w_j C_j$	WSPT	$\frac{1}{2}(1+\sqrt{2})$	(Kawaguchi, Kyan '86)

まとめと次回の予告

目標

リスト・スケジューリング を例にして
近似アルゴリズムの解析手法 を身に付ける

- 重要概念：近似保証，近似比
- 対象とする問題：$P \mid\mid C_{\max}$，$P \mid\mid \sum C_j$，$P \mid\mid \sum w_j C_j$

これで前半が終了

次回の予告

先行制約 (precedence constraint) の扱い