

情報数理科学特別講義 D / 情報数理科学特別講義 H / 情報数理科学特殊講義 B  
第 3 回  
近似アプローチ：貪欲法

岡本 吉央  
okamotoy@uec.ac.jp

電気通信大学

2012 年 8 月 27 日

最終更新：2012 年 8 月 29 日 11:34

- ① 近似アルゴリズム
- ② 貪欲法
- ③ 例 1 : 巡回セールスマン問題
- ④ 例 2 : 同一機械並列スケジューリング
- ⑤ 第 3 回のまとめと補足

## アルゴリズムとは

### アルゴリズムがすること (簡単に言うと)

- ▶ データを受け取り (入力)
- ▶ それを処理し (処理)
- ▶ 結果を返す (出力)

### アルゴリズムに要求されること

- ▶ どんな入力に対しても (入力の普遍性)
- ▶ 高速に処理し (処理の高速性)
- ▶ 正しい結果を返す (出力の正当性)

この3つの要求を緩和できる場合がある

## 緩和の仕方：「出力の正当性」の緩和

## アルゴリズムに要求されること

- ▶ どんな入力に対しても (入力の普遍性)
- ▶ 高速に処理し (処理の高速性)
- ▶ 正しい結果を返す (出力の正当性)

## 「出力の正当性」の緩和

- ▶ どんな入力に対しても (入力の普遍性)
- ▶ 高速に処理し (処理の高速性)
- ▶ 近似的な結果を返す (出力の正当性の緩和)

## 近似比：最小化問題の場合

## 定義：近似比 (approximation ratio)

最小化問題に対するアルゴリズムの**近似比**とは、任意の入力に対して

$$\text{OPT} \leq \text{ALG} \leq \alpha \cdot \text{OPT}$$

を満たす  $\alpha \geq 1$  のこと。ただし、

- ▶ OPT：最適値
- ▶ ALG：アルゴリズムの出力値

## 注

- ▶ 「 $\text{OPT} \leq \text{ALG}$ 」はどんなアルゴリズムでも成り立つ
- ▶ 近似比が小さいほど、(近似の意味で) 性能のよいアルゴリズム

## 近似比：最大化問題の場合

## 定義：近似比 (approximation ratio)

最大化問題に対するアルゴリズムの**近似比**とは、任意の入力に対して

$$\text{ALG} \leq \text{OPT} \leq \alpha \cdot \text{ALG}$$

を満たす  $\alpha \geq 1$  のこと。ただし、

- ▶ OPT：最適値
- ▶ ALG：アルゴリズムの出力値

## 注

- ▶ 「 $\text{ALG} \leq \text{OPT}$ 」はどんなアルゴリズムでも成り立つ
- ▶ 近似比が小さいほど、(近似の意味で) 性能のよいアルゴリズム

- ① 近似アルゴリズム
- ② 貪欲法
- ③ 例 1 : 巡回セールスマン問題
- ④ 例 2 : 同一機械並列スケジューリング
- ⑤ 第 3 回のまとめと補足

# 貪欲法 (貪欲アルゴリズム) とは？

## 貪欲法 (greedy algorithm, greedy method)

一番「得をする」要素から順に選ぶことで解を構築する方法

## 今日の目標

### 次の理解

- ▶ 近似比の理解
- ▶ 貪欲法の典型例を理解
- ▶ 貪欲法の性能解析を理解



- ① 近似アルゴリズム
- ② 貪欲法
- ③ 例 1 : 巡回セールスマン問題
- ④ 例 2 : 同一機械並列スケジューリング
- ⑤ 第 3 回のまとめと補足

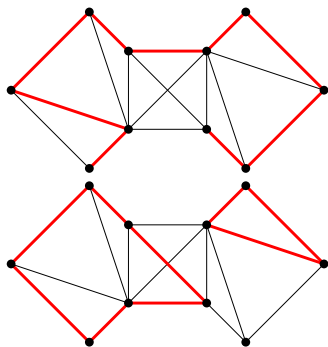
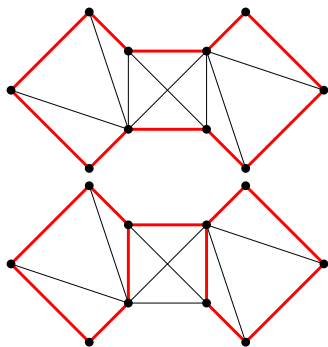
## 巡回路 (ハミルトン閉路)

 $G = (V, E)$ : 無向グラフ

定義: 巡回路 (ハミルトン閉路)

 $G$  の巡回路とは, すべての頂点をちょうど一度ずつ通る閉路

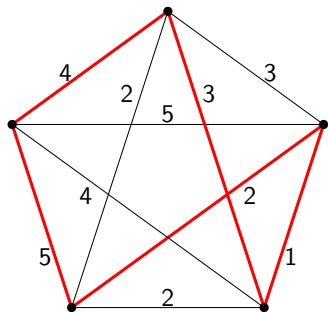
どれが巡回路?



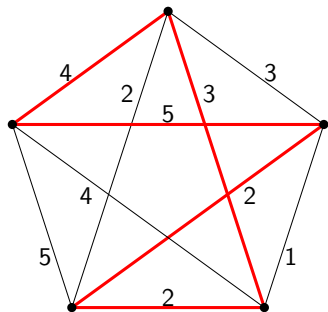
## 巡回セールスマン問題

## 定義: 巡回セールスマン問題

- ▶ 入力: **完全**グラフ  $G = (V, E)$ , 各辺  $e$  に対する費用  $c(e) \geq 0$
- ▶ 出力:  $G$  の巡回路
- ▶ 目的: 巡回路に使われている辺の費用和の最小化



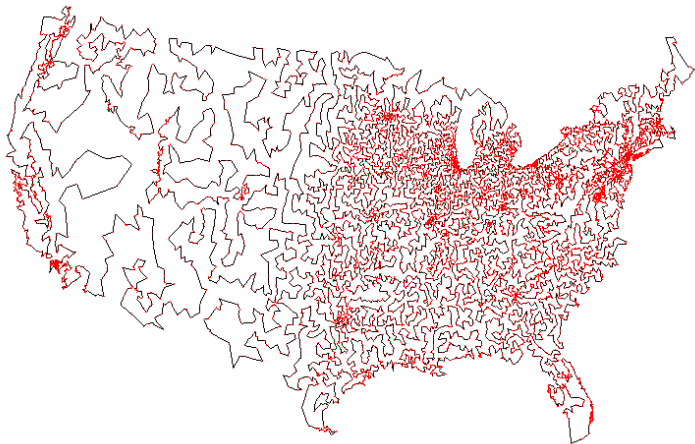
費用和 = 15



費用和 = 16

## 巡回セールスマン問題: 別の例

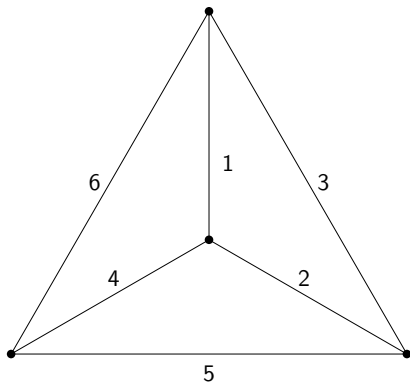
頂点は平面上の点, 辺の費用は2点間のユークリッド距離 (直線距離)



[http://www.tsp.gatech.edu/usa13509/usa13509\\_sol.html](http://www.tsp.gatech.edu/usa13509/usa13509_sol.html)

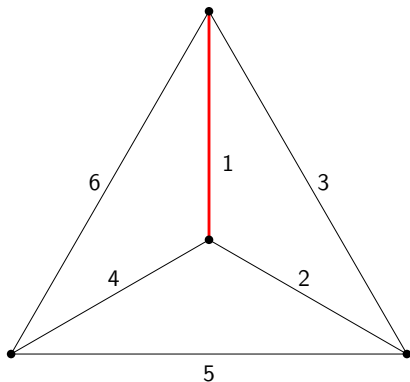
## 巡回セールスマン問題に対する貪欲法

- 1  $T \leftarrow \emptyset$  とする ( $T$  が最終的に出力する巡回路となる)
- 2 費用の小さい順に辺  $e$  を見ていく
- 3  $T$  に  $e$  を加えることで巡回路が作れなくなることがないならば,  $T$  に  $e$  を加える
- 4  $T$  が巡回路ならば,  $T$  を出力して停止. そうでなければ 2 に戻る



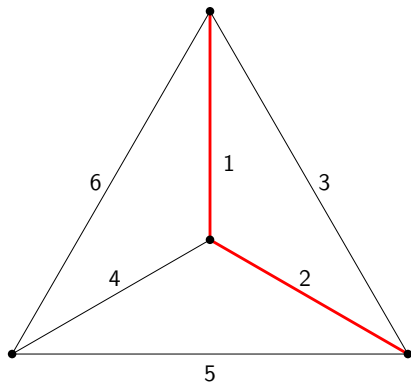
## 巡回セールスマン問題に対する貪欲法

- 1  $T \leftarrow \emptyset$  とする ( $T$  が最終的に出力する巡回路となる)
- 2 費用の小さい順に辺  $e$  を見ていく
- 3  $T$  に  $e$  を加えることで巡回路が作れなくなることがないならば,  $T$  に  $e$  を加える
- 4  $T$  が巡回路ならば,  $T$  を出力して停止. そうでなければ 2 に戻る



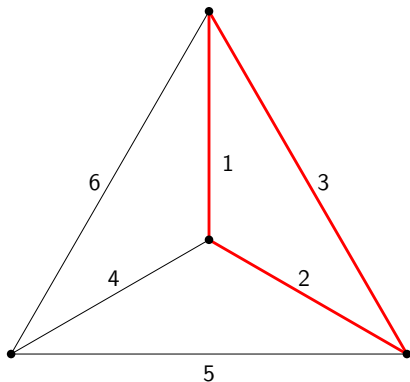
## 巡回セールスマン問題に対する貪欲法

- 1  $T \leftarrow \emptyset$  とする ( $T$  が最終的に出力する巡回路となる)
- 2 費用の小さい順に辺  $e$  を見ていく
- 3  $T$  に  $e$  を加えることで巡回路が作れなくなることがないならば,  $T$  に  $e$  を加える
- 4  $T$  が巡回路ならば,  $T$  を出力して停止. そうでなければ 2 に戻る



## 巡回セールスマン問題に対する貪欲法

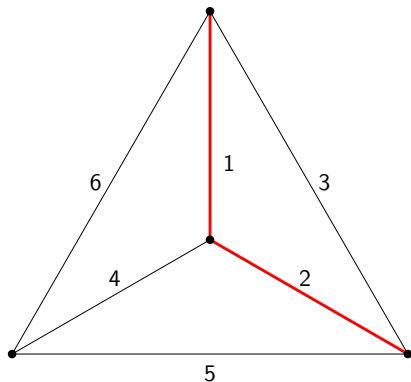
- 1  $T \leftarrow \emptyset$  とする ( $T$  が最終的に出力する巡回路となる)
- 2 費用の小さい順に辺  $e$  を見ていく
- 3  $T$  に  $e$  を加えることで巡回路が作れなくなることがないならば,  $T$  に  $e$  を加える
- 4  $T$  が巡回路ならば,  $T$  を出力して停止. そうでなければ 2 に戻る





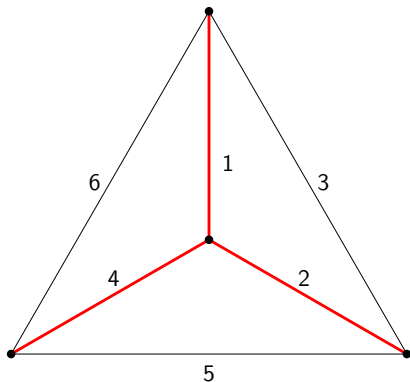
## 巡回セールスマン問題に対する貪欲法

- 1  $T \leftarrow \emptyset$  とする ( $T$  が最終的に出力する巡回路となる)
- 2 費用の小さい順に辺  $e$  を見ていく
- 3  $T$  に  $e$  を加えることで巡回路が作れなくなることがないならば,  $T$  に  $e$  を加える
- 4  $T$  が巡回路ならば,  $T$  を出力して停止. そうでなければ 2 に戻る



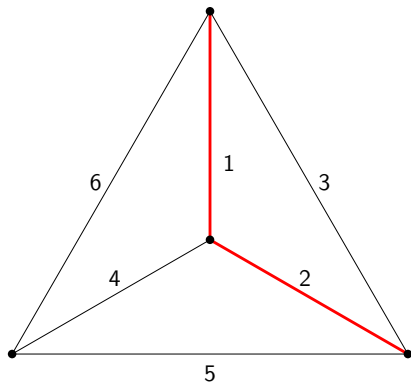
## 巡回セールスマン問題に対する貪欲法

- 1  $T \leftarrow \emptyset$  とする ( $T$  が最終的に出力する巡回路となる)
- 2 費用の小さい順に辺  $e$  を見ていく
- 3  $T$  に  $e$  を加えることで巡回路が作れなくなることがないならば,  $T$  に  $e$  を加える
- 4  $T$  が巡回路ならば,  $T$  を出力して停止. そうでなければ 2 に戻る



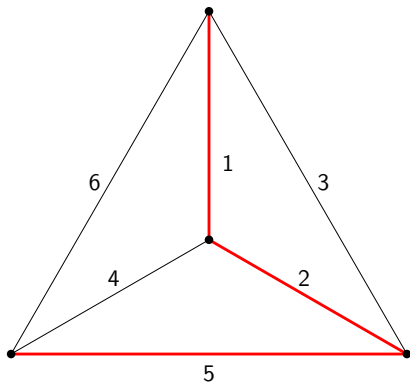
## 巡回セールスマン問題に対する貪欲法

- 1  $T \leftarrow \emptyset$  とする ( $T$  が最終的に出力する巡回路となる)
- 2 費用の小さい順に辺  $e$  を見ていく
- 3  $T$  に  $e$  を加えることで巡回路が作れなくなることがないならば,  $T$  に  $e$  を加える
- 4  $T$  が巡回路ならば,  $T$  を出力して停止. そうでなければ 2 に戻る



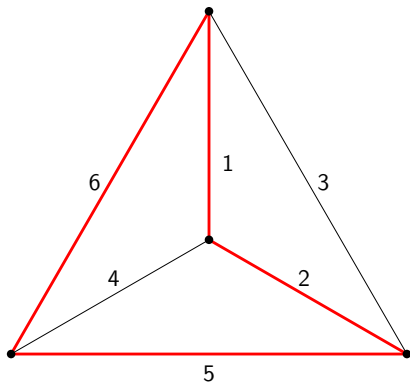
## 巡回セールスマン問題に対する貪欲法

- 1  $T \leftarrow \emptyset$  とする ( $T$  が最終的に出力する巡回路となる)
- 2 費用の小さい順に辺  $e$  を見ていく
- 3  $T$  に  $e$  を加えることで巡回路が作れなくなることがないならば,  $T$  に  $e$  を加える
- 4  $T$  が巡回路ならば,  $T$  を出力して停止. そうでなければ 2 に戻る



## 巡回セールスマン問題に対する貪欲法

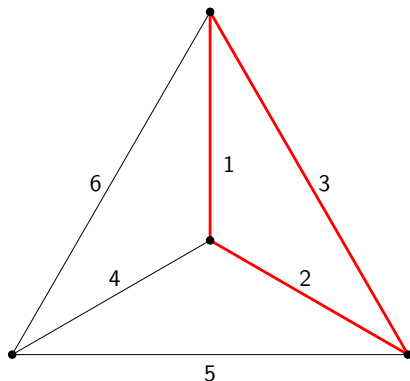
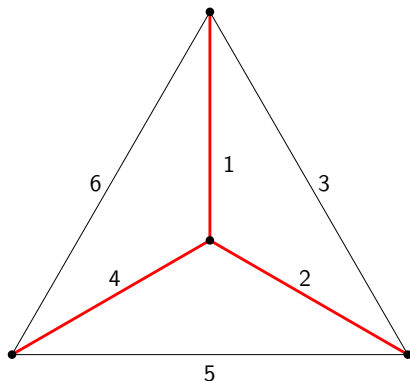
- 1  $T \leftarrow \emptyset$  とする ( $T$  が最終的に出力する巡回路となる)
- 2 費用の小さい順に辺  $e$  を見ていく
- 3  $T$  に  $e$  を加えることで巡回路が作れなくなることがないならば,  $T$  に  $e$  を加える
- 4  $T$  が巡回路ならば,  $T$  を出力して停止. そうでなければ 2 に戻る



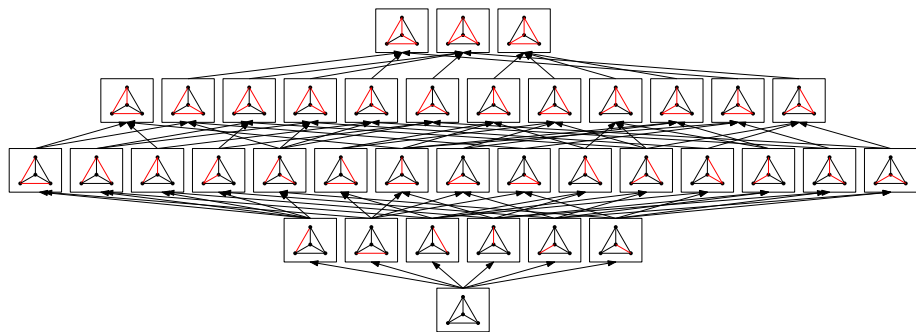
## 巡回セールスマン問題に対する貪欲法: 注意

「巡回路が作れなくなることはない」とは?

- ▶ どの頂点に接続する辺の数も 3 以上ではない
- ▶ 一部の頂点のみを通る閉路が存在しない



## 巡回セールスマン問題に対する貪欲法 : 山登り

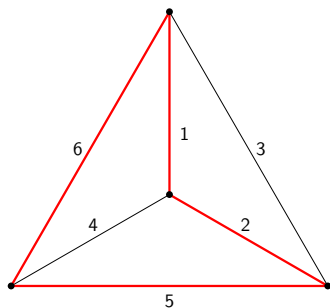


## アルゴリズム設計で最初に確認すべきこと

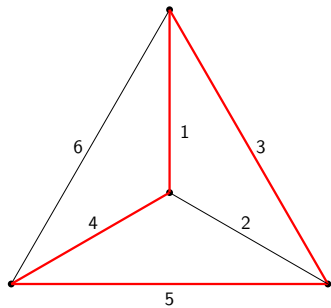
- ▶ 考えたアルゴリズムは必ず最適解を出力するのか？
  - ▶ 考えたアルゴリズムが最適解を出力しない入力はあるのか？
- ▶ 考えたアルゴリズムの近似比はどれほどなのか？
  - ▶ 考えたアルゴリズムの近似比が悪いことを示す入力はあるのか？



## 貪欲法が最適巡回回路を出力しない例



$$\text{ALG} = 14$$

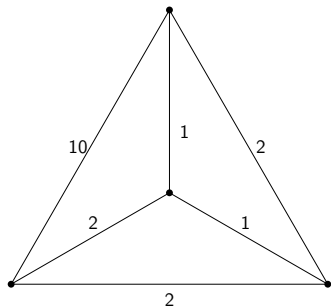


$$\text{OPT} \leq 13$$

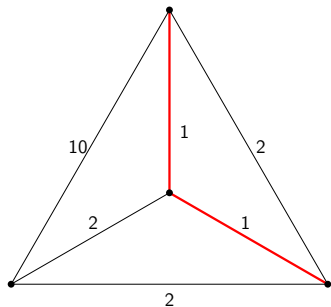
この例は

- ▶ 「貪欲法が最適巡回回路を出力しない」ことを示している  
 $\text{ALG} > \text{OPT}$  ( $\because \text{ALG} = 14 > 13 \geq \text{OPT}$ )
- ▶ 「貪欲法の近似比が  $14/13$  より良くはない」ことを示している  
 $\text{ALG} \geq \frac{14}{13} \text{OPT}$  ( $\because \text{ALG} = 14 = \frac{14}{13} \cdot 13 \geq \frac{14}{13} \text{OPT}$ )

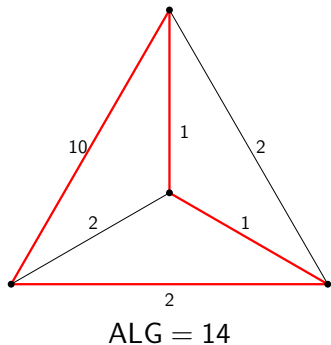
## 貪欲法の近似比が悪いことを示す例



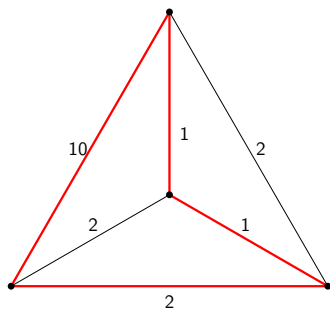
## 貪欲法の近似比が悪いことを示す例



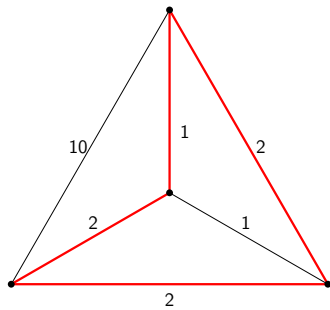
## 貪欲法の近似比が悪いことを示す例



## 貪欲法の近似比が悪いことを示す例



$$\text{ALG} = 14$$



$$\text{OPT} \leq 7$$

この例は

- ▶ 「貪欲法の近似比が 2 より良くはない」ことを示している

$$\text{ALG} \geq 2\text{OPT}$$

$$(\because \text{ALG} = 14 = \frac{14}{7} \cdot 7 \geq 2\text{OPT})$$

## 貪欲法の近似比は抑えられない

## 定理 1 (貪欲法の近似比の下界)

任意の定数  $\alpha \geq 1$  に対して,  
巡回セールスマン問題に対する貪欲法の近似比は  $\alpha$  より良くはない

証明

- ▶ 与えられた任意の  $\alpha \geq 1$  に対して  
貪欲法の出力する巡回路の費用 ALG と最適巡回路の費用 OPT が

$$ALG \geq \alpha \cdot OPT$$

を満たすような問題例を作成すればよい

- ▶ 実際に作成する (板書で)

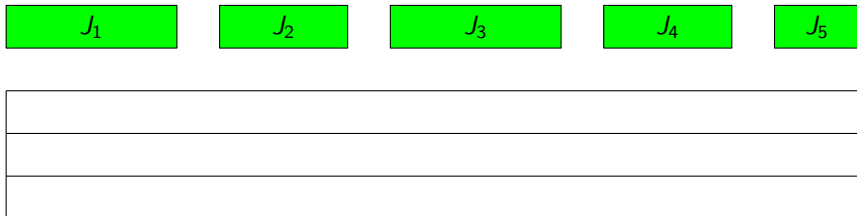


- ① 近似アルゴリズム
- ② 貪欲法
- ③ 例 1 : 巡回セールスマン問題
- ④ 例 2 : 同一機械並列スケジューリング
- ⑤ 第 3 回のまとめと補足

## 同一機械並列スケジューリング問題

## 定義: 同一機械並列スケジューリング問題

- ▶ 入力:  $m$  個の同一な機械  $M_1, \dots, M_m$ ,  $n$  個のジョブ  $J_1, \dots, J_n$ , ジョブ  $J_i$  を機械で処理するための時間  $p_i > 0$
- ▶ 出力: 機械へのジョブの割当
- ▶ 目的: ジョブ処理がすべて完了するまでの時間 (最終終了時刻) の最小化

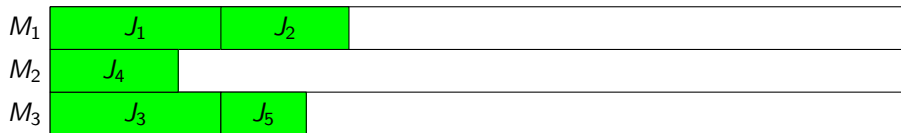




## 同一機械並列スケジューリング問題

## 定義: 同一機械並列スケジューリング問題

- ▶ 入力:  $m$  個の同一な機械  $M_1, \dots, M_m$ ,  $n$  個のジョブ  $J_1, \dots, J_n$ , ジョブ  $J_i$  を機械で処理するための時間  $p_i > 0$
- ▶ 出力: 機械へのジョブの割当
- ▶ 目的: ジョブ処理がすべて完了するまでの時間 (最終終了時刻) の最小化



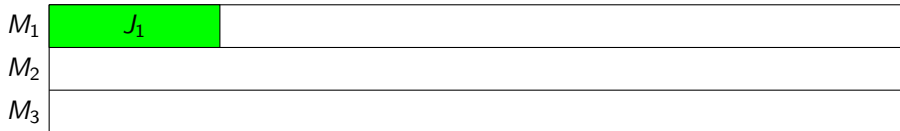
## 同一機械並列スケジューリング問題に対する貪欲法

- 1  $A_1, \dots, A_m \leftarrow \emptyset$  とする ( $A_j$  が  $M_j$  で処理するジョブ集合となる)
- 2  $i = 1, \dots, n$  の順にジョブ  $J_i$  を見ていく
- 3 機械の中で割り当てられたジョブの処理時間之和が最小の機械  $M_j$  を見つける (複数ある場合は, 添字の最も小さいものとする)
- 4  $A_j \leftarrow A_j \cup \{J_i\}$  とする
- 5  $i = n$  ならば  $A_1, \dots, A_m$  を出力して停止. そうでなければ, 2 に戻る

 $J_1$  $J_2$  $J_3$  $J_4$  $J_5$  $M_1$  $M_2$  $M_3$

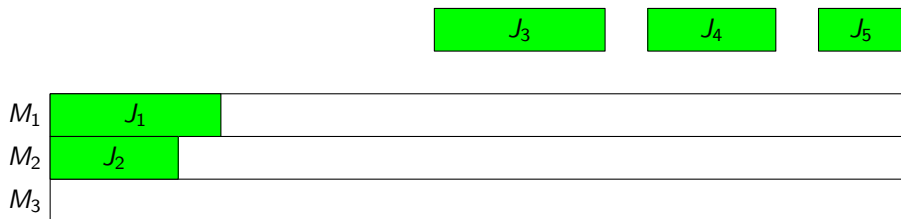
## 同一機械並列スケジューリング問題に対する貪欲法

- 1  $A_1, \dots, A_m \leftarrow \emptyset$  とする ( $A_j$  が  $M_j$  で処理するジョブ集合となる)
- 2  $i = 1, \dots, n$  の順にジョブ  $J_i$  を見ていく
- 3 機械の中で割り当てられたジョブの処理時間之和が最小の機械  $M_j$  を見つける (複数ある場合は, 添字の最も小さいものとする)
- 4  $A_j \leftarrow A_j \cup \{J_i\}$  とする
- 5  $i = n$  ならば  $A_1, \dots, A_m$  を出力して停止. そうでなければ, 2 に戻る



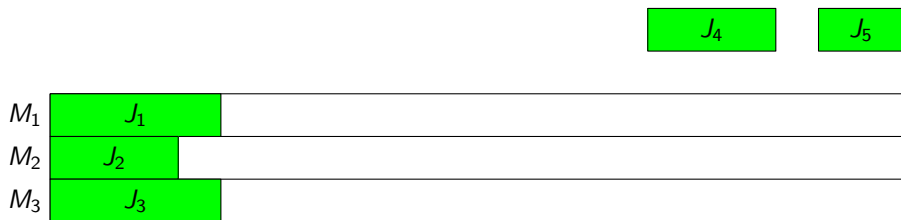
## 同一機械並列スケジューリング問題に対する貪欲法

- 1  $A_1, \dots, A_m \leftarrow \emptyset$  とする ( $A_j$  が  $M_j$  で処理するジョブ集合となる)
- 2  $i = 1, \dots, n$  の順にジョブ  $J_i$  を見ていく
- 3 機械の中で割り当てられたジョブの処理時間之和が最小の機械  $M_j$  を見つける (複数ある場合は, 添字の最も小さいものとする)
- 4  $A_j \leftarrow A_j \cup \{J_i\}$  とする
- 5  $i = n$  ならば  $A_1, \dots, A_m$  を出力して停止. そうでなければ, 2 に戻る



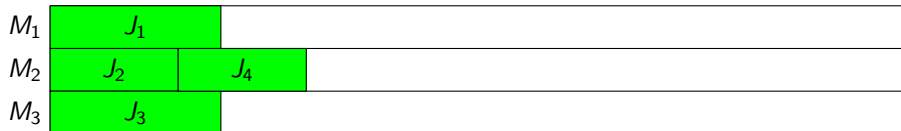
## 同一機械並列スケジューリング問題に対する貪欲法

- 1  $A_1, \dots, A_m \leftarrow \emptyset$  とする ( $A_j$  が  $M_j$  で処理するジョブ集合となる)
- 2  $i = 1, \dots, n$  の順にジョブ  $J_i$  を見ていく
- 3 機械の中で割り当てられたジョブの処理時間之和が最小の機械  $M_j$  を見つける (複数ある場合は, 添字の最も小さいものとする)
- 4  $A_j \leftarrow A_j \cup \{J_i\}$  とする
- 5  $i = n$  ならば  $A_1, \dots, A_m$  を出力して停止. そうでなければ, 2 に戻る



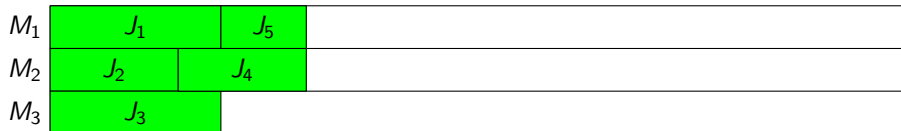
## 同一機械並列スケジューリング問題に対する貪欲法

- 1  $A_1, \dots, A_m \leftarrow \emptyset$  とする ( $A_j$  が  $M_j$  で処理するジョブ集合となる)
- 2  $i = 1, \dots, n$  の順にジョブ  $J_i$  を見ていく
- 3 機械の中で割り当てられたジョブの処理時間之和が最小の機械  $M_j$  を見つける (複数ある場合は, 添字の最も小さいものとする)
- 4  $A_j \leftarrow A_j \cup \{J_i\}$  とする
- 5  $i = n$  ならば  $A_1, \dots, A_m$  を出力して停止. そうでなければ, 2 に戻る

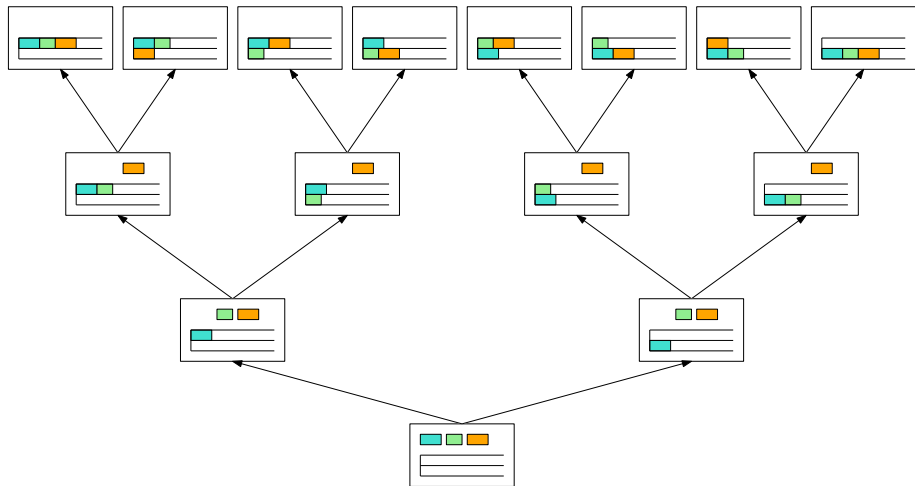



## 同一機械並列スケジューリング問題に対する貪欲法

- 1  $A_1, \dots, A_m \leftarrow \emptyset$  とする ( $A_j$  が  $M_j$  で処理するジョブ集合となる)
- 2  $i = 1, \dots, n$  の順にジョブ  $J_i$  を見ていく
- 3 機械の中で割り当てられたジョブの処理時間之和が最小の機械  $M_j$  を見つける (複数ある場合は, 添字の最も小さいものとする)
- 4  $A_j \leftarrow A_j \cup \{J_i\}$  とする
- 5  $i = n$  ならば  $A_1, \dots, A_m$  を出力して停止. そうでなければ, 2 に戻る



# 同一機械並列スケジューリングに対する貪欲法：山登り

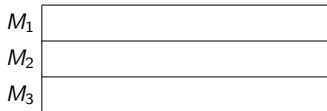




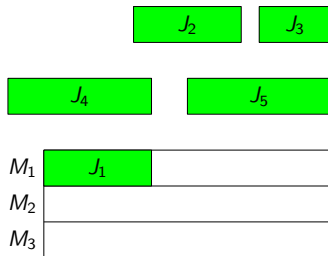
## アルゴリズム設計で最初に確認すべきこと (再掲)

- ▶ 考えたアルゴリズムは必ず最適解を出力するのか？
  - ▶ 考えたアルゴリズムが最適解を出力しない入力はあるのか？
- ▶ 考えたアルゴリズムの近似比はどれほどなのか？
  - ▶ 考えたアルゴリズムの近似比が悪いことを示す入力はあるのか？

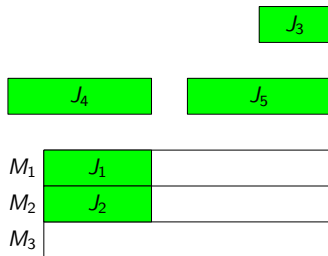
## 貪欲法が最適解を出力しない例



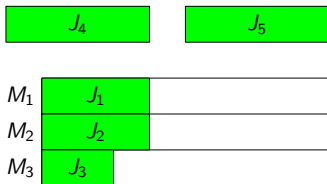
## 貪欲法が最適解を出力しない例



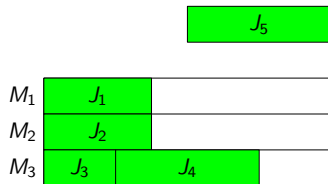
## 貪欲法が最適解を出力しない例



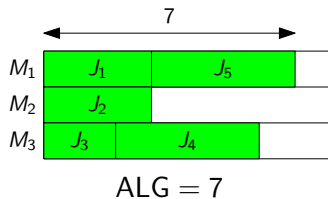
## 貪欲法が最適解を出力しない例



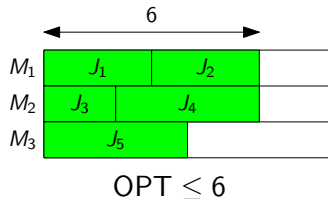
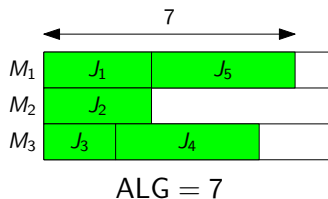
## 貪欲法が最適解を出力しない例



## 貪欲法が最適解を出力しない例



## 貪欲法が最適解を出力しない例

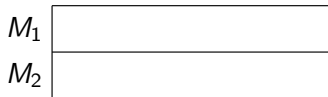


この例は

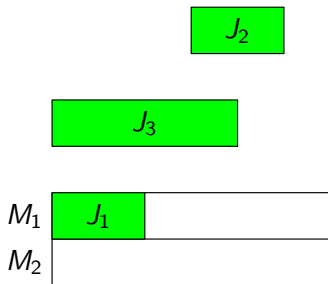
- ▶ 「貪欲法が最適スケジュールを出力しない」ことを示している  
 $ALG > OPT$  ( $\because ALG = 7 > 6 \geq OPT$ )
- ▶ 「貪欲法の近似比が  $7/6$  より良くはない」ことを示している  
 $ALG \geq \frac{7}{6}OPT$  ( $\because ALG = 7 = \frac{7}{6} \cdot 6 \geq \frac{7}{6}OPT$ )



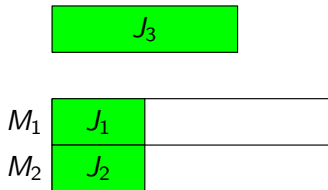
## 貪欲法の近似比が悪いことを示す例

 $J_1$  $J_2$  $J_3$ 

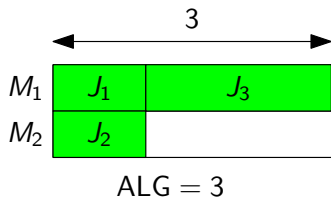
## 貪欲法の近似比が悪いことを示す例



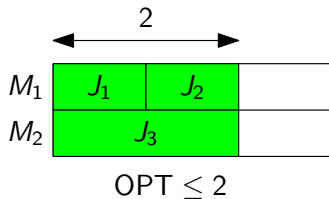
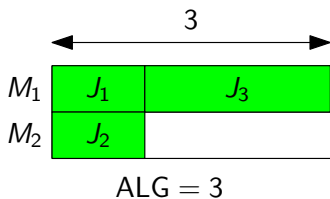
## 貪欲法の近似比が悪いことを示す例



## 貪欲法の近似比が悪いことを示す例



## 貪欲法の近似比が悪いことを示す例



この例は

- ▶ 「貪欲法の近似比が  $3/2$  より良くはない」ことを示している

$$\text{ALG} \geq \frac{3}{2} \text{OPT}$$

$$(\because \text{ALG} = 3 = \frac{3}{2} \cdot 2 \geq \frac{3}{2} \text{OPT})$$

## 貪欲法の近似比保証

## 定理 2

(Graham '66)

同一機械並列スケジューリング問題に対する貪欲法の近似比は  $2 - \frac{1}{m}$  以下。  
すなわち、任意の入力に対して、  
貪欲法の出力が与える最終完了時刻 ALG と最適最終完了時刻 OPT に

$$\text{ALG} \leq \left(2 - \frac{1}{m}\right) \cdot \text{OPT}$$

という関係が成り立つ

つまり、 $m = 2$  のとき、

- ▶ 貪欲法の近似比は  $3/2$  以下 (定理 2)
- ▶ 貪欲法の近似比は  $3/2$  より良くはない (前のスライドの例)
- ▶  $\therefore$  貪欲法の近似比は  $3/2$  であり、これは **タイト** (tight)

## 貪欲法の近似比保証：証明 (1)

## 証明

- ▶ 次の3つを証明する：完了が最も遅いジョブを  $J_i$  とする
  - 1  $p_i \leq \text{OPT}$
  - 2  $m \cdot \text{OPT} \geq p_1 + p_2 + \dots + p_n$
  - 3  $m \cdot (\text{ALG} - p_i) \leq p_1 + p_2 + \dots + p_{i-1} + p_{i+1} + \dots + p_n$
- ▶ この3つが証明できれば，定理 2 も証明できる (板書)

## 証明すべきこと

- 1  $p_i \leq \text{OPT}$

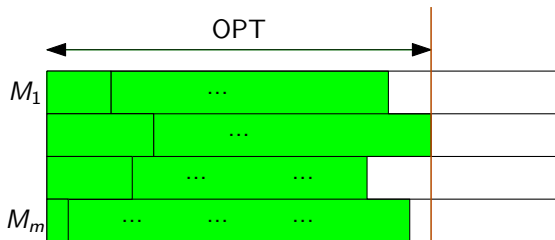
証明：これは簡単 (どうして?) □

## 貪欲法の近似比保証: 証明 (2)

## 証明すべきこと

$$2 \ m \cdot \text{OPT} \geq p_1 + p_2 + \dots + p_n$$

証明: OPT を与えている状況を表す次の図を見る □





## 貪欲法の近似比保証：証明 (3)

## 証明すべきこと

$$3 \quad m \cdot (\text{ALG} - p_i) \leq p_1 + p_2 + \dots + p_{i-1} + p_{i+1} + \dots + p_n$$

証明：ALG を与えている状況を表す次の図を見る



- ▶  $J_i$  の処理の開始時刻に，どの機械も必ずジョブを処理している  
(どうして?)
- ▶ そこから，示したい式が得られる □

## 貪欲法の近似比保証：証明 (3)

## 証明すべきこと

$$3 \quad m \cdot (\text{ALG} - p_i) \leq p_1 + p_2 + \dots + p_{i-1} + p_{i+1} + \dots + p_n$$

証明：ALG を与えている状況を表す次の図を見る



- ▶  $J_i$  の処理の開始時刻に，どの機械も必ずジョブを処理している  
(どうして?)
- ▶ そこから，示したい式が得られる □

## 貪欲法の近似比保証: 証明 (3)

## 証明すべきこと

$$3 \quad m \cdot (\text{ALG} - p_i) \leq p_1 + p_2 + \dots + p_{i-1} + p_{i+1} + \dots + p_n$$

証明: ALG を与えている状況を表す次の図を見る



- ▶  $J_i$  の処理の開始時刻に, どの機械も必ずジョブを処理している (どうして?)
- ▶ そこから, 示したい式が得られる

これで定理 2 の証明が完了した

## 第 3 回のまとめ

近似比  $\alpha$ 

- ▶  $OPT \leq ALG \leq \alpha \cdot OPT$  (最小化問題)
- ▶  $ALG \leq OPT \leq \alpha \cdot ALG$  (最大化問題)
- ▶  $\alpha \geq 1$  が小さいほど, (近似の意味で) よいアルゴリズム

## 貪欲法とその例

- ▶ 巡回セールスマン問題
- ▶ 同一機械並列スケジューリング

## 近似アルゴリズム設計で最初に確認すべきこと

- ▶ 考えたアルゴリズムが最適解を出力しない入力はあるのか？
- ▶ 考えたアルゴリズムの近似比が悪いことを示す入力はあるのか？

## 補足：巡回セールスマン問題

## 貪欲法の近似比

- ▶ 費用が三角不等式を満たす場合
  - ▶ 任意の入力に対して  $O(\log n)$  以下 (Frieze '78)
  - ▶ ある入力に対して  $\Omega(\log n / \log \log n)$  以上 (Frieze '78)

## 他の多項式時間アルゴリズムの近似率

- ▶ 費用が三角不等式を満たす場合
  - ▶ どのような入力に対しても  $3/2$  以下 (Christofides '76)
- ▶ 頂点が平面上の点，費用が 2 点間のユークリッド距離に対応する場合
  - ▶ どのような入力に対しても  $1 + \epsilon$  以下 (Arora '98, Mitchell '99)

## 補足：同一機械並列スケジューリング問題

## 他の多項式時間アルゴリズムの近似比

- ▶  $\frac{4}{3} - \frac{1}{3m}$  以下 (Graham '66)
  - ▶ ジョブを処理時間の長い順に並べ直してから，貪欲法を適用  
(演習問題参照)
- ▶  $1 + \epsilon$  以下 ( $m$  が定数であるとき) (Lenstra, Shmoys, Tardos '90)

## 参考文献

## ▶ 近似アルゴリズム一般

- ▶ V. V. Vazirani. Approximation Algorithms. Corrected 2nd printing, Springer, 2001.  
(日本語訳：浅野孝夫訳「近似アルゴリズム」，丸善出版，2002.)
- ▶ D.P. Williamson and D.B. Shmoys. The Design of Approximation Algorithms. Cambridge University Press, 2011.
- ▶ D.S. Hochbaum (ed.). Approximation Algorithms for NP-Hard Problems. PWS Publishing Company, 1997.

## ▶ 巡回セールスマン問題一般

- ▶ The Traveling Salesman Problem. <http://www.tsp.gatech.edu/>
- ▶ E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, D.B. Shmoys. The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization, John Wiley & Sons, 1985.
- ▶ 山本芳嗣，久保幹雄．巡回セールスマン問題への招待．朝倉書店，1997．

## 参考文献

## ▶ 参考文献

- ▶ S. Arora. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. *Journal of the ACM* (1998) 2-11.
- ▶ N. Christofides. Worst case analysis of a new heuristic for the traveling salesman problem. Report 388, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, PA, 1976.
- ▶ A.M. Frieze. Worst-case analysis of algorithms for traveling salesman problems. *Operations Research Verfahren* 32 (1978) 94-112.
- ▶ R. Graham. Bounds for certain multiprocessing anomalies. *Bell System Technical Journal* 45 (1966) 1563-1581.
- ▶ J.K. Lenstra, D.B. Shmoys, É. Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical Programming* 46 (1990) 259-271.
- ▶ J.S.B. Mitchell. Guillotine subdivisions approximate polygonal subdivisions: A simple polynomial-time approximation scheme for geometric TSP,  $k$ -MST, and related problems. *SIAM Journal on Computing* 28 (1999) 1298-1309.



# 目次

- ① 近似アルゴリズム
- ② 貪欲法
- ③ 例 1：巡回セールスマン問題
- ④ 例 2：同一機械並列スケジューリング
- ⑤ 第 3 回のまとめと補足