I482F 実践的アルゴリズム特論
(9) 近似アルゴリズム：貪欲法

岡本 吉央
okamotoy@jaist.ac.jp

北陸先端科学技術大学院大学

2011 年 6 月 18 日

”**最終更新：**2011/06/17 13:40”

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)

「処理の高速性」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

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

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

- 「$\mathrm{OPT} \leq \mathrm{ALG}$」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

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

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

- **「$\mathrm{ALG} \leq \mathrm{OPT}$」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**

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**

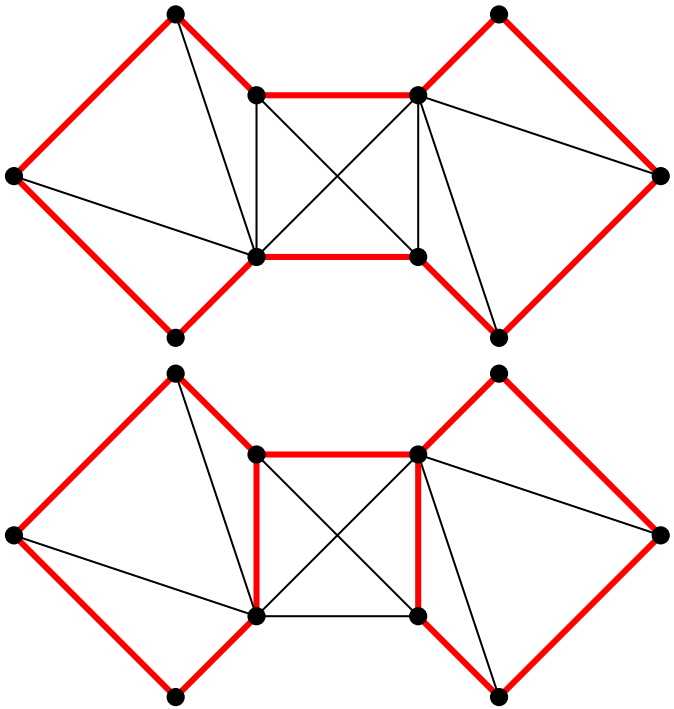
巡回路 (ハミルトン閉路)

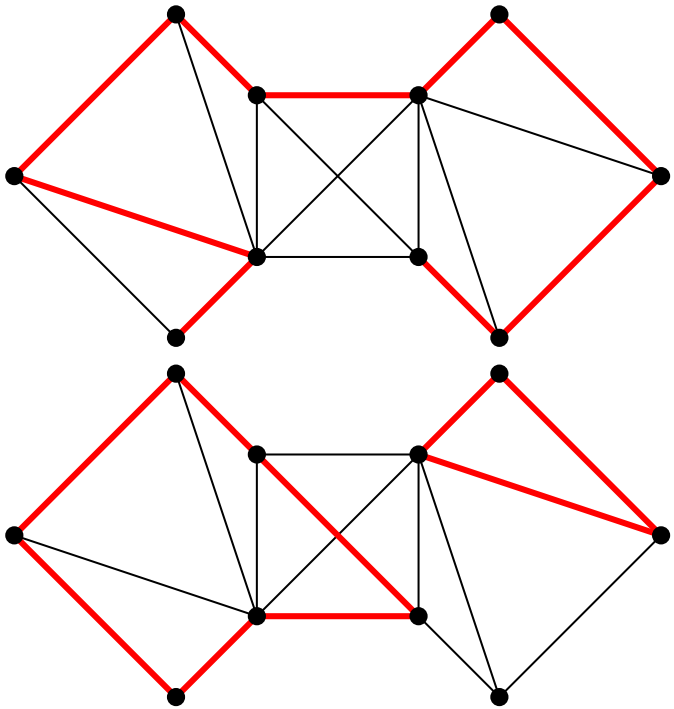
$G = (V, E)$：無向グラフ

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？

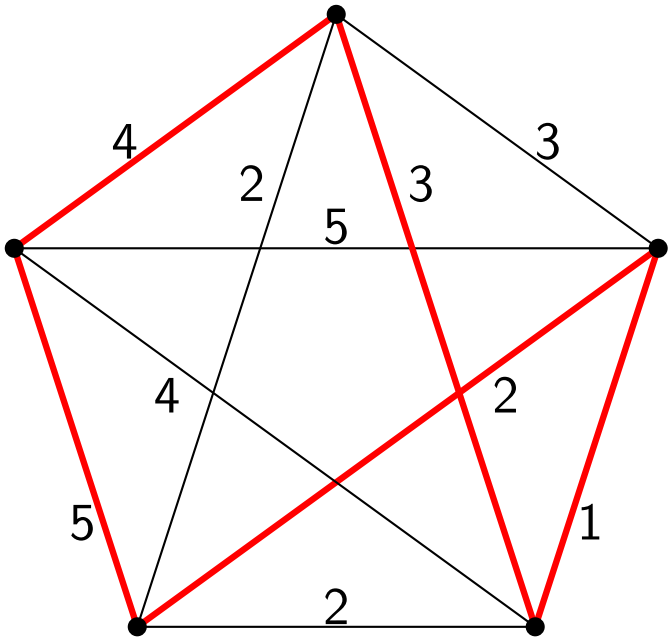




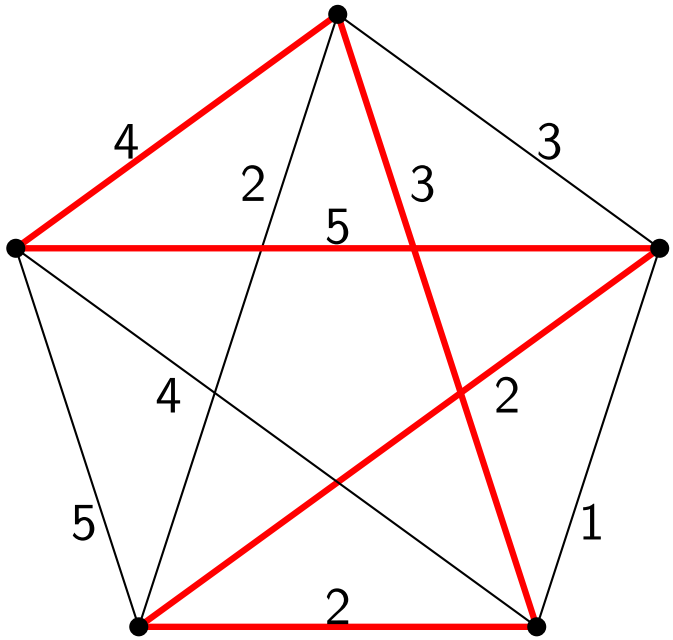
巡回セールスマン問題

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



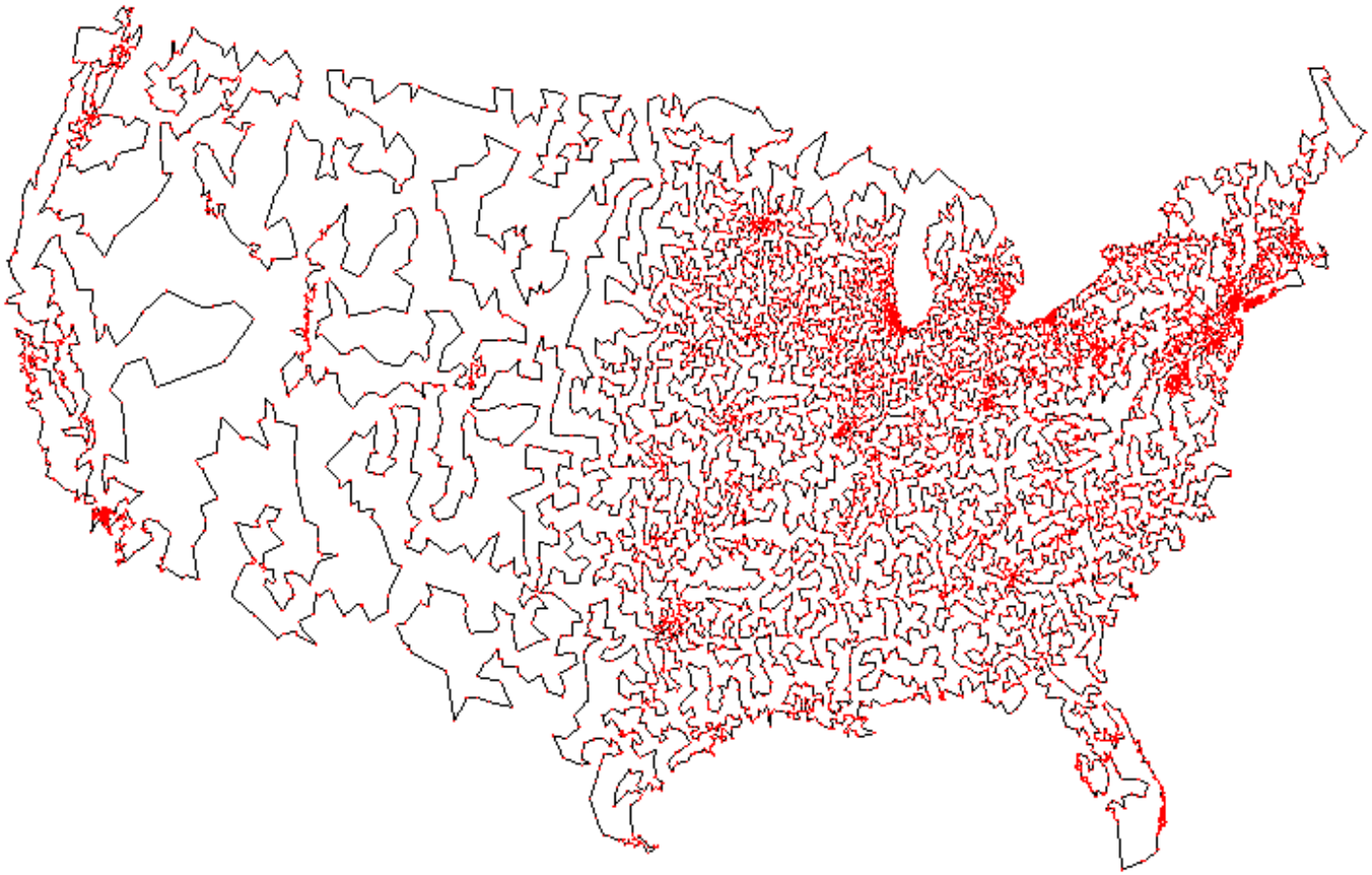
費用和 $= 15$



費用和 $= 16$

巡回セールスマン問題：別の例

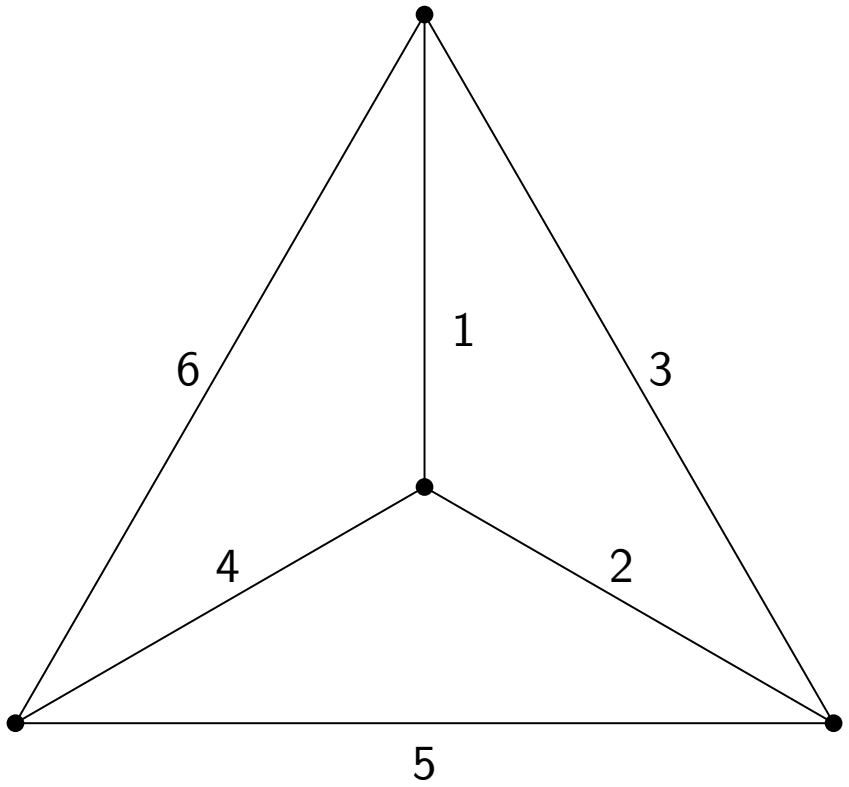
頂点は平面上の点，辺の費用は 2 点間のユークリッド距離 (直線距離)



http://www.tsp.gatech.edu/usa13509/usa13509_sol.html

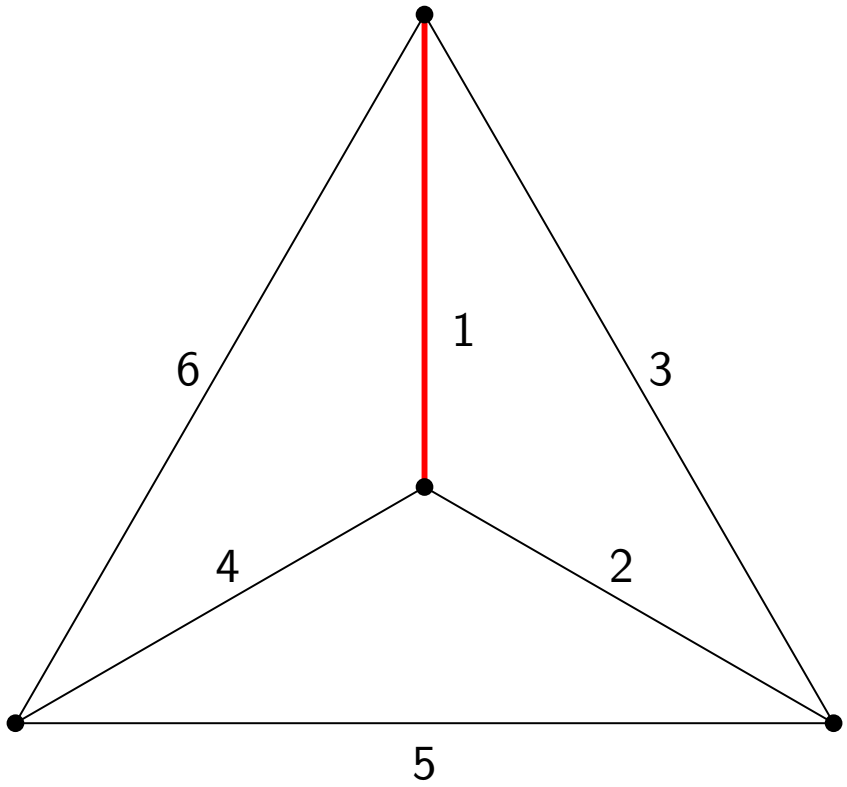
巡回セールスマン問題に対する貪欲法

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



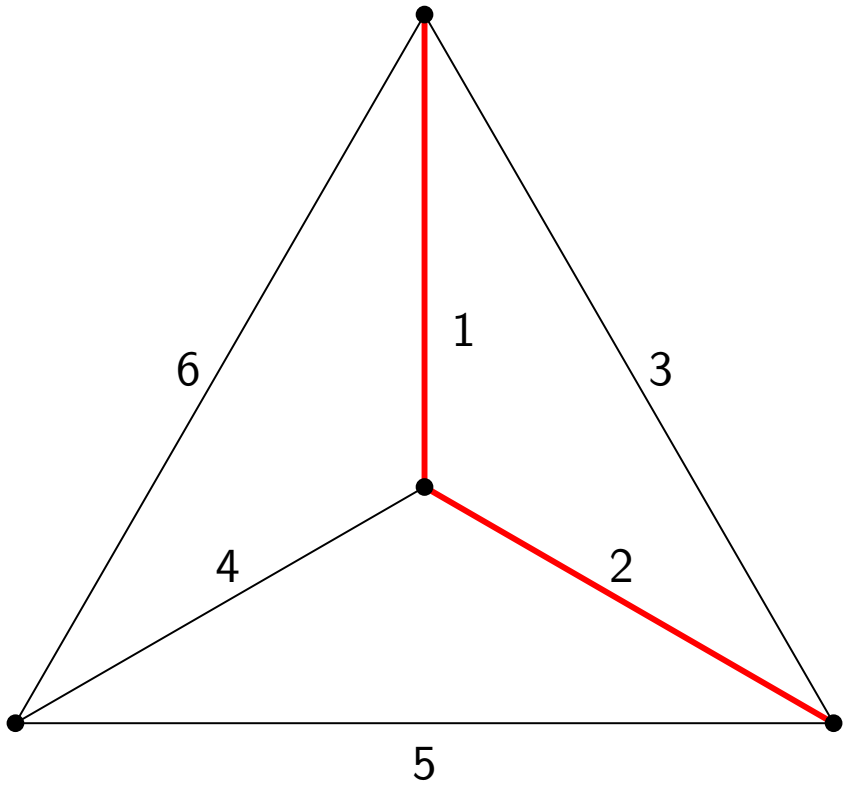
巡回セールスマン問題に対する貪欲法

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



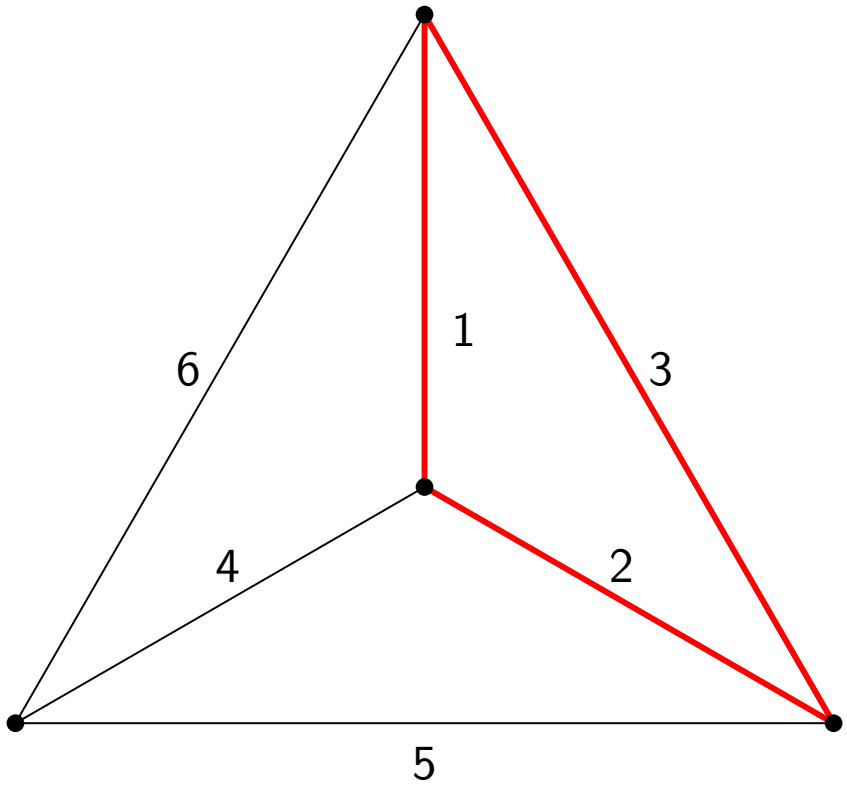
巡回セールスマン問題に対する貪欲法

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



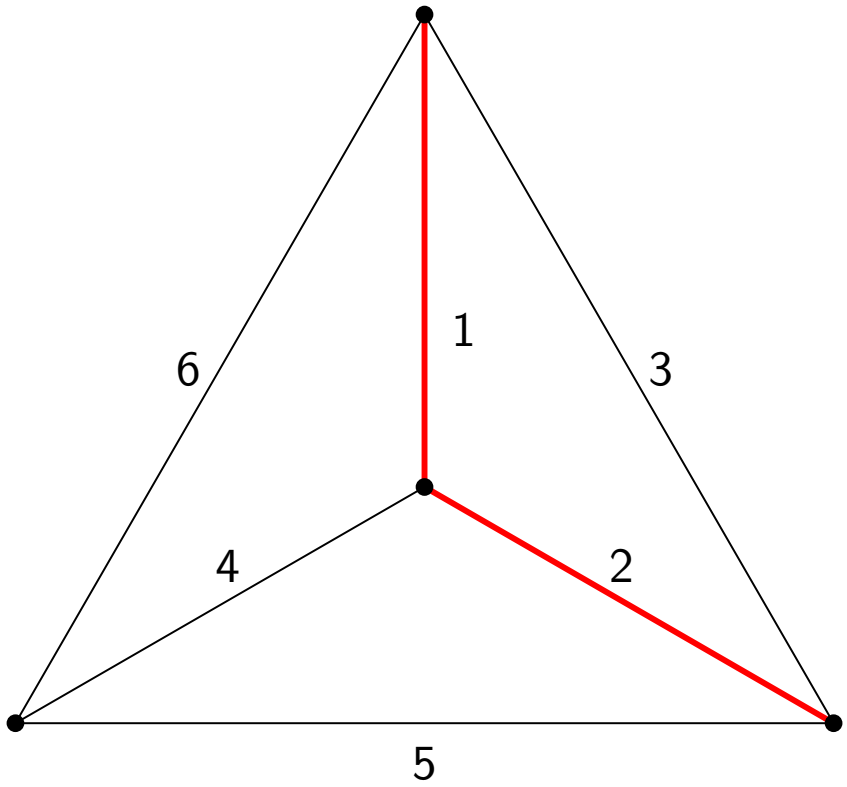
巡回セールスマン問題に対する貪欲法

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



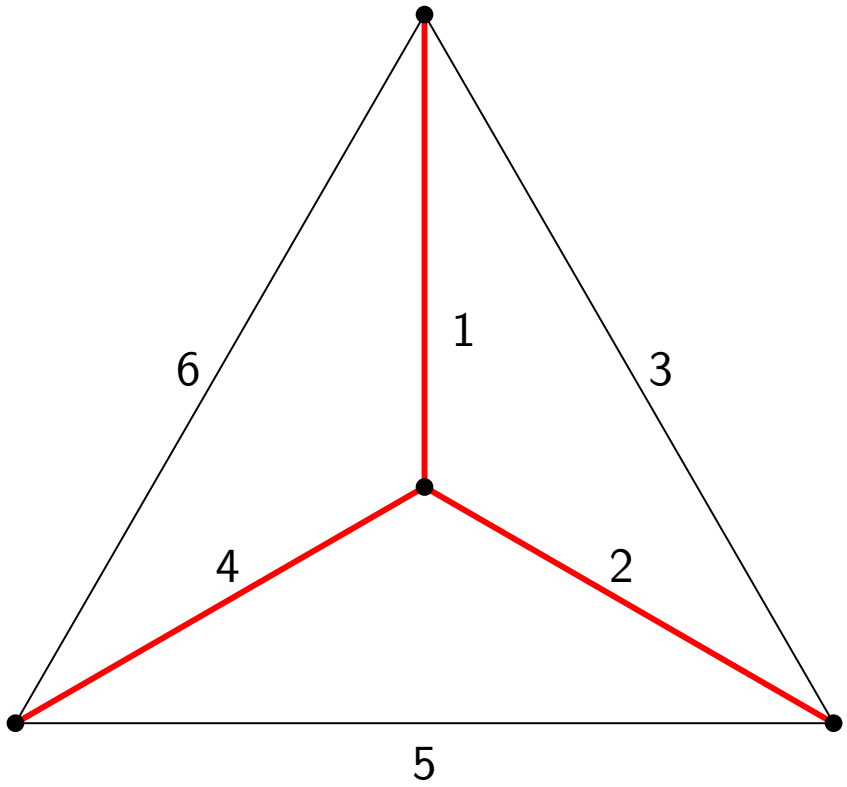
巡回セールスマン問題に対する貪欲法

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



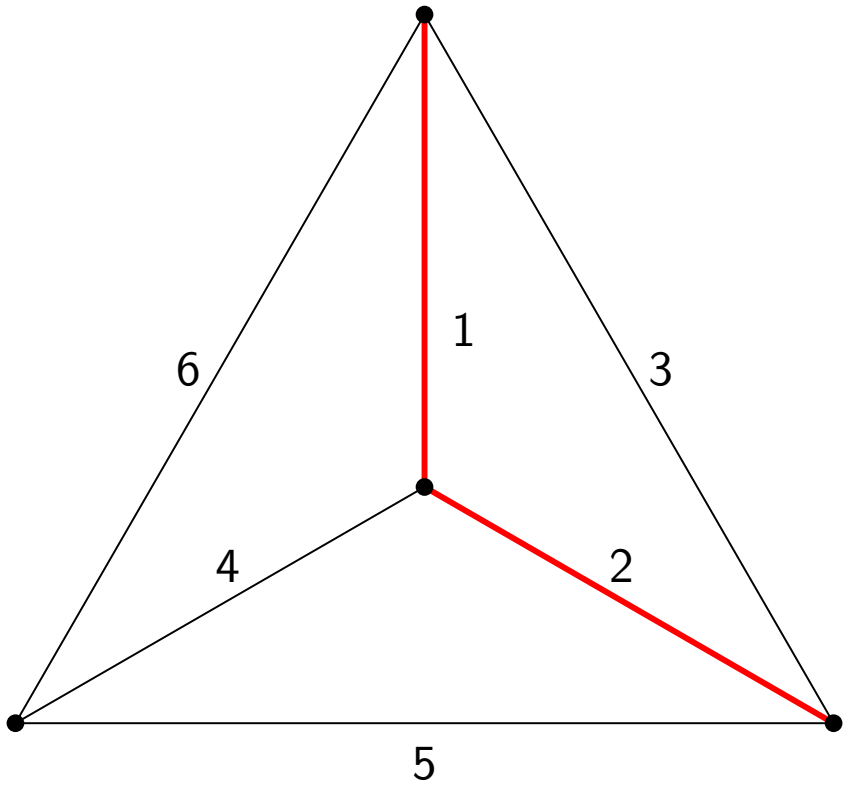
巡回セールスマン問題に対する貪欲法

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



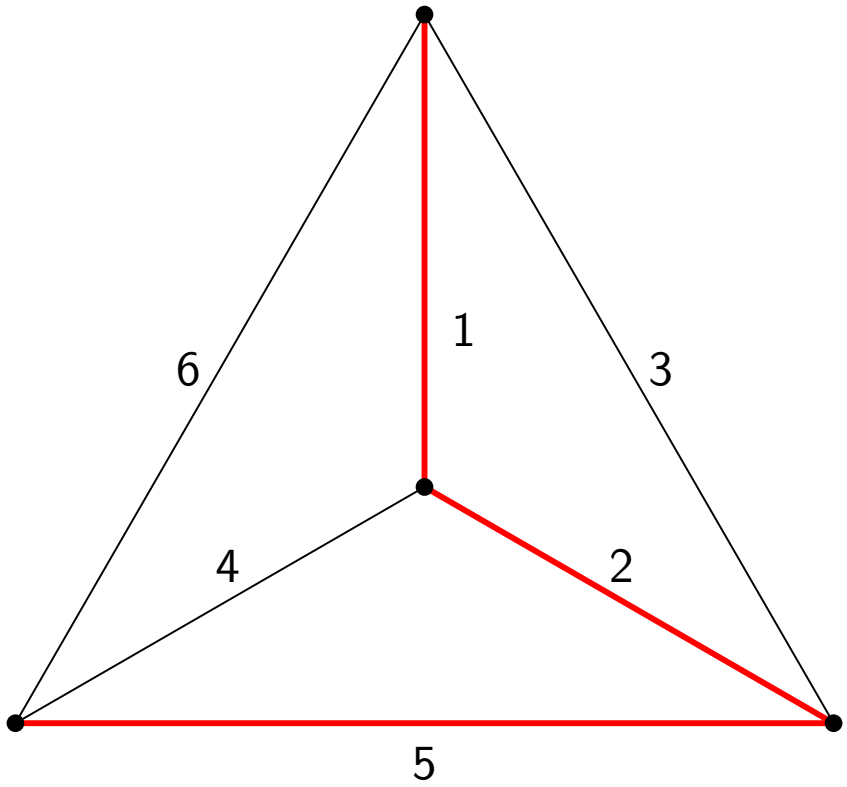
巡回セールスマン問題に対する貪欲法

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



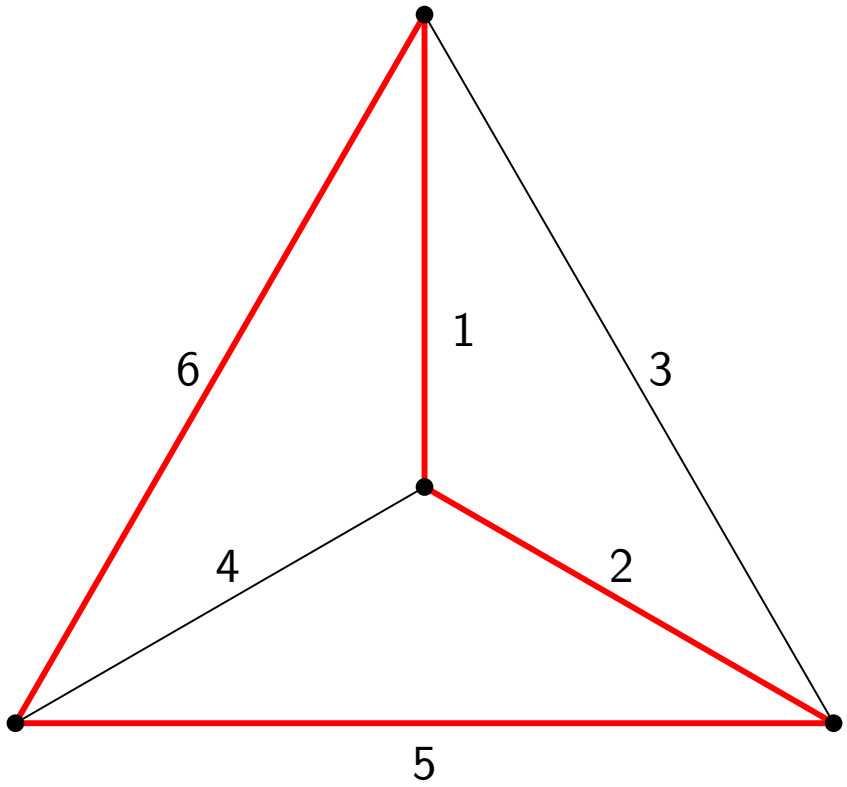
巡回セールスマン問題に対する貪欲法

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

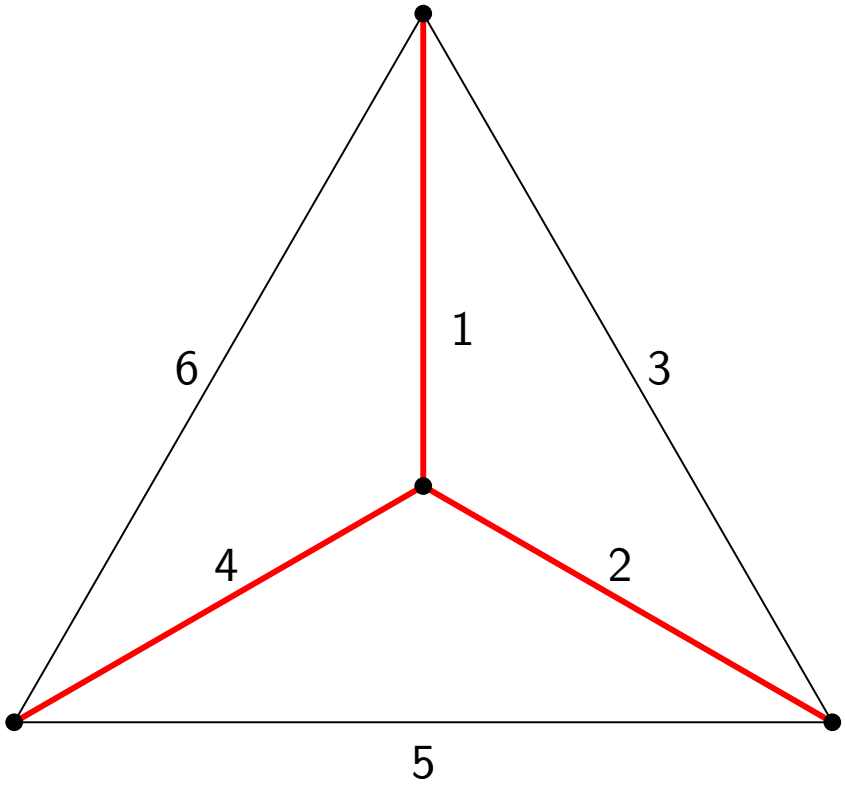
1. $T \leftarrow \emptyset$ とする (T が最終的に出力する巡回路となる)
2. 費用の小さい順に辺 e を見ていく
3. T に e を加えることで巡回路が作れなくなることがないならば，T に e を加える
4. T が巡回路ならば，T を出力して停止．そうでなければ 2 に戻る

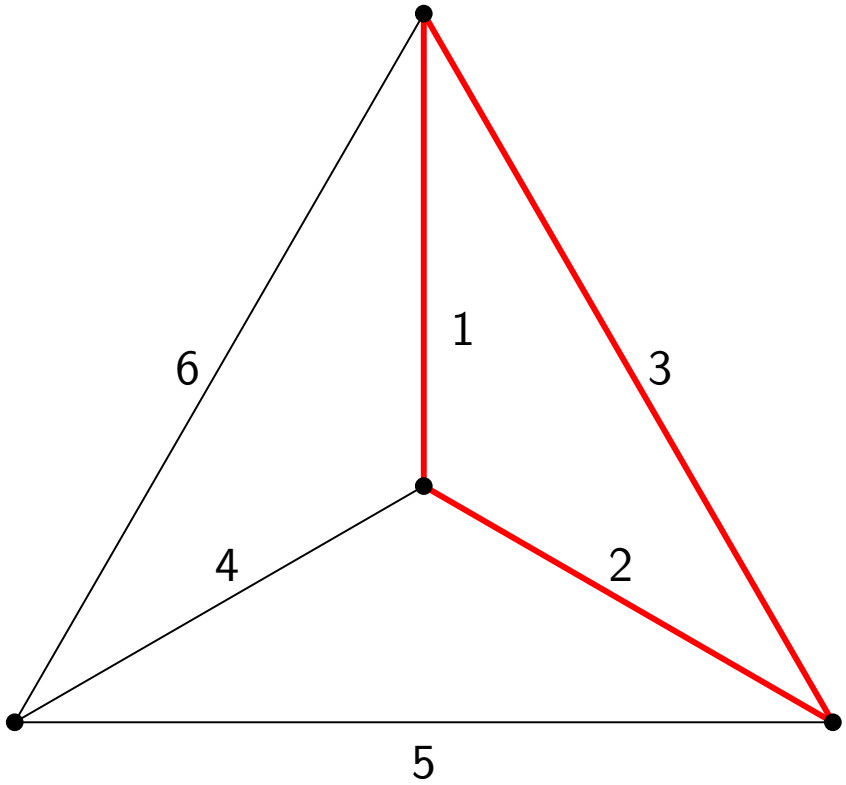


巡回セールスマン問題に対する貪欲法：注意

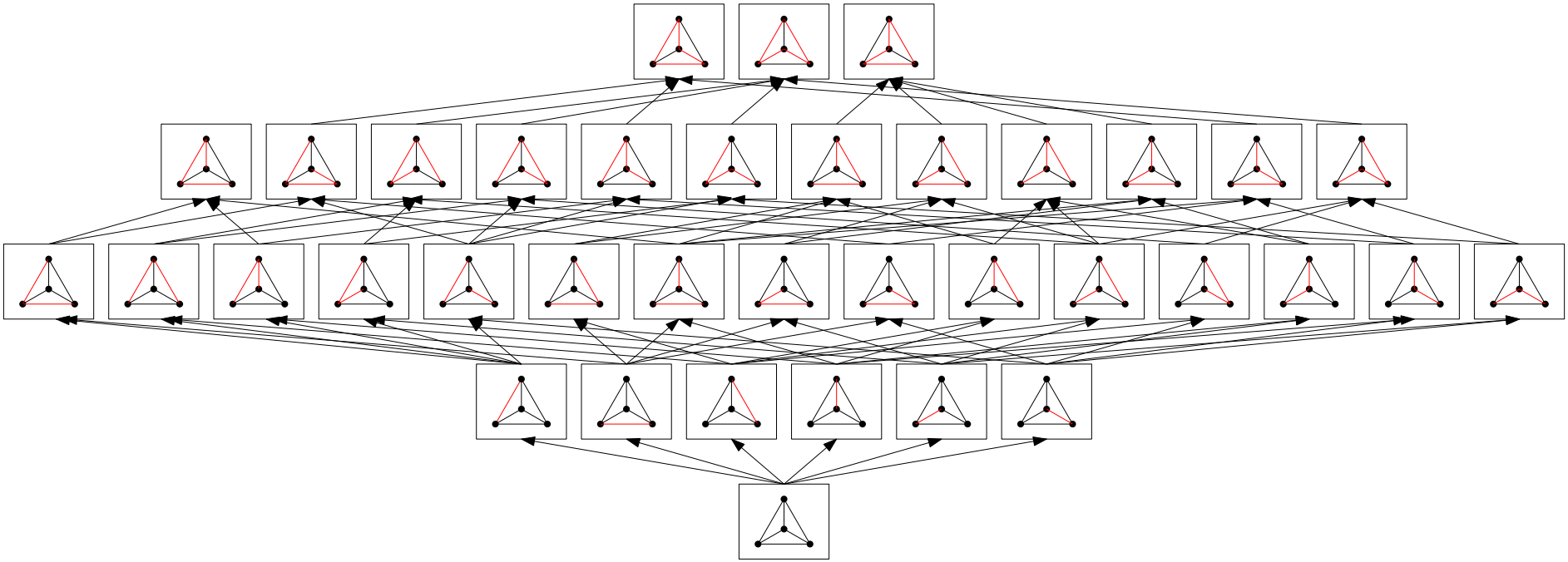
「巡回路が作れなくなることがない」とは？

- 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





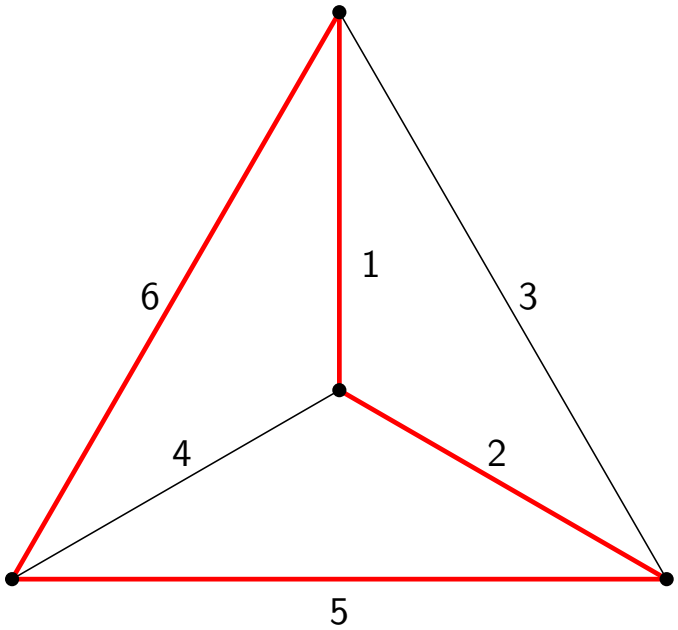
巡回セールスマン問題に対する貪欲法：山登り



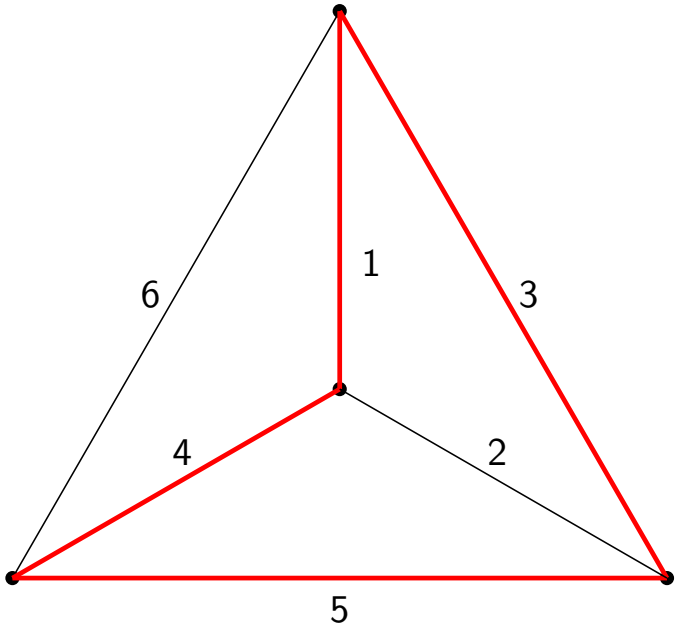
アルゴリズム設計で最初に確認するべきこと

- 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？

貪欲法が最適巡回路を出力しない例



$\mathrm{ALG} = 14$

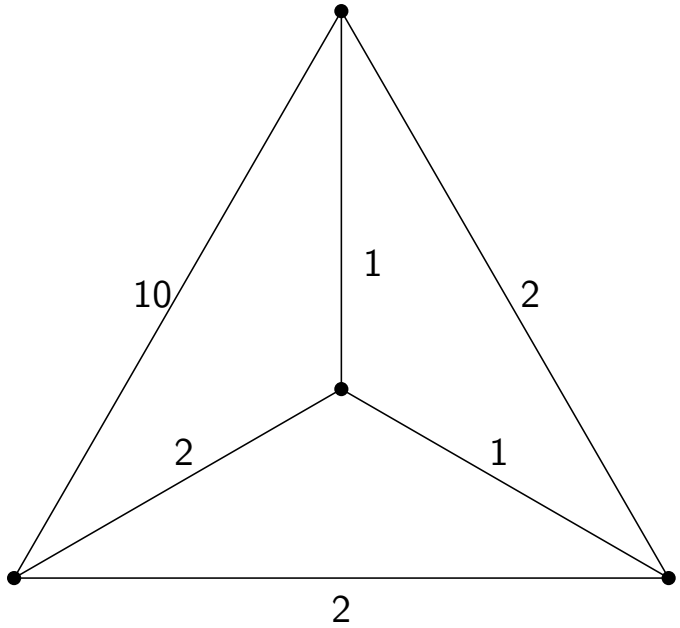


$\mathrm{OPT} \leq 13$

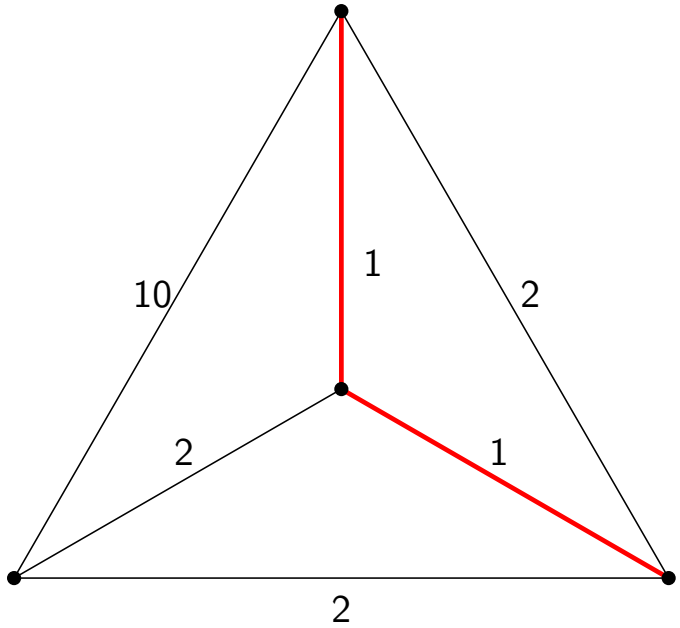
この例は

- **「貪欲法が最適巡回路を出力しない」ことを示している**
  $\mathrm{ALG} > \mathrm{OPT}$ $\quad (\because \mathrm{ALG} = 14 > 13 \geq \mathrm{OPT})$
- **「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**
  $\mathrm{ALG} \geq \frac{14}{13}\mathrm{OPT}$ $\quad (\because \mathrm{ALG} = 14 = \frac{14}{13} \cdot 13 \geq \frac{14}{13}\mathrm{OPT})$

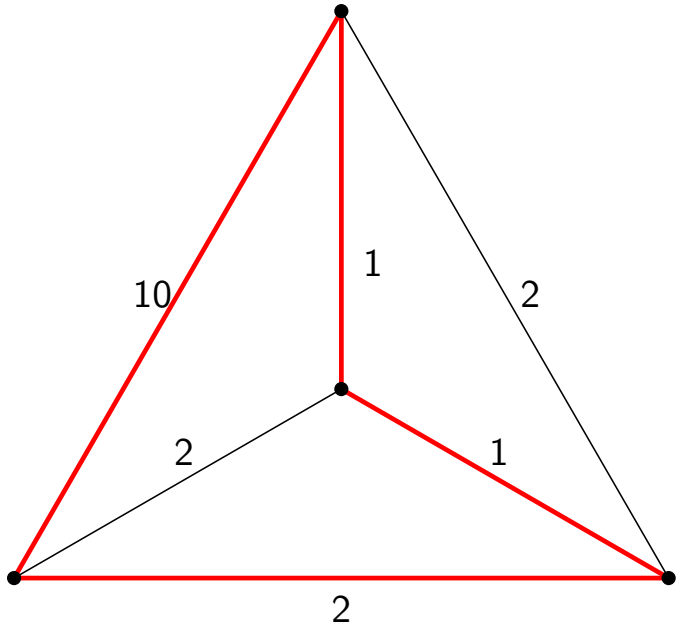
貪欲法の近似比が悪いことを示す例



貪欲法の近似比が悪いことを示す例

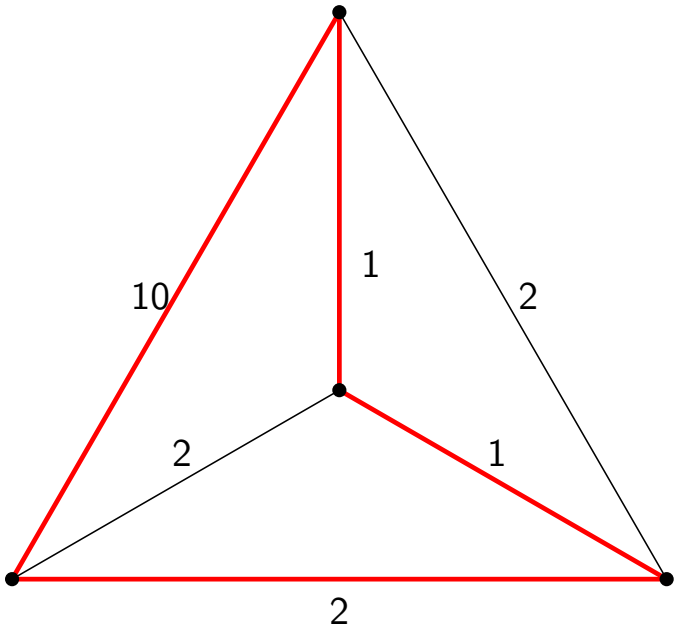


貪欲法の近似比が悪いことを示す例

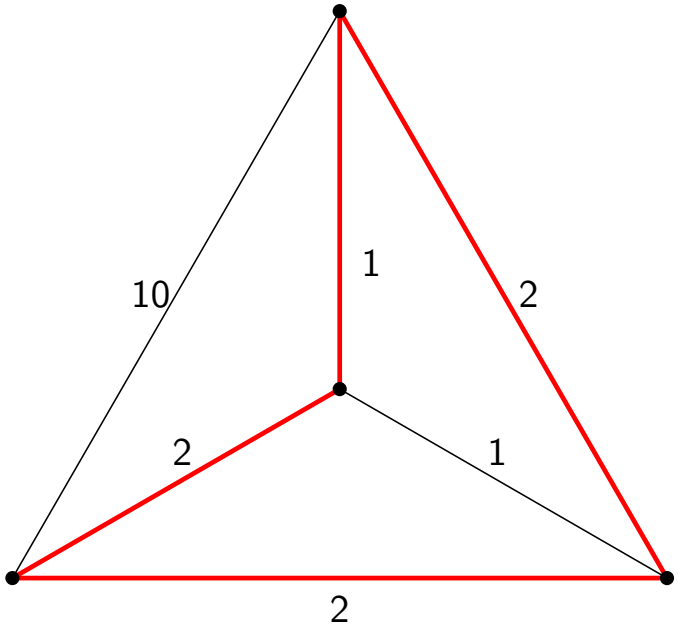


$$\mathrm{ALG} = 14$$

貪欲法の近似比が悪いことを示す例



ALG = 14



OPT ≤ 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**

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

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 α 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

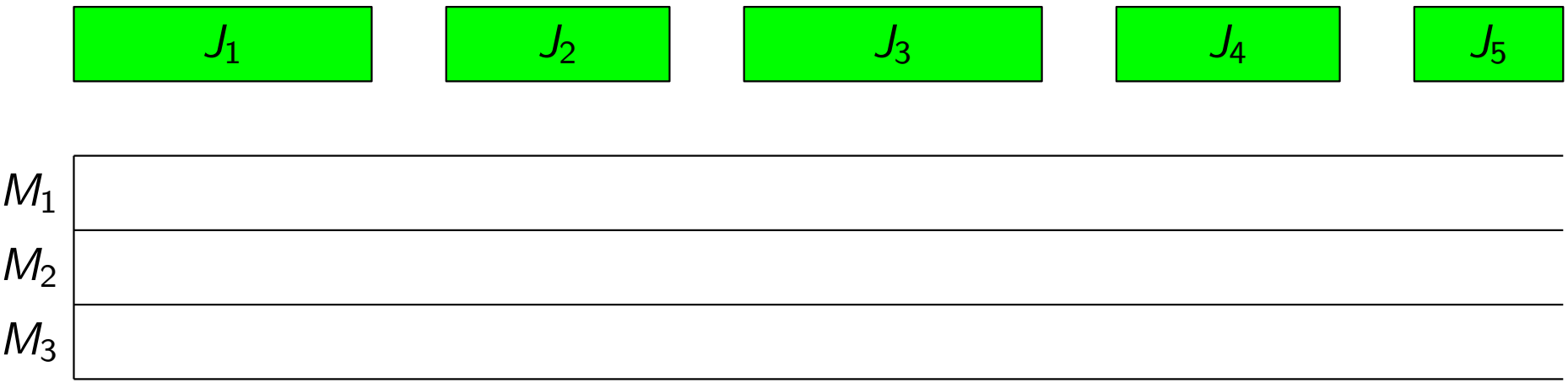
$$\mathrm{ALG} \geq \alpha \cdot \mathrm{OPT}$$

  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)** □

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 $M_1, \ldots, M_m$，n 個のジョブ $J_1, \ldots, J_n$，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



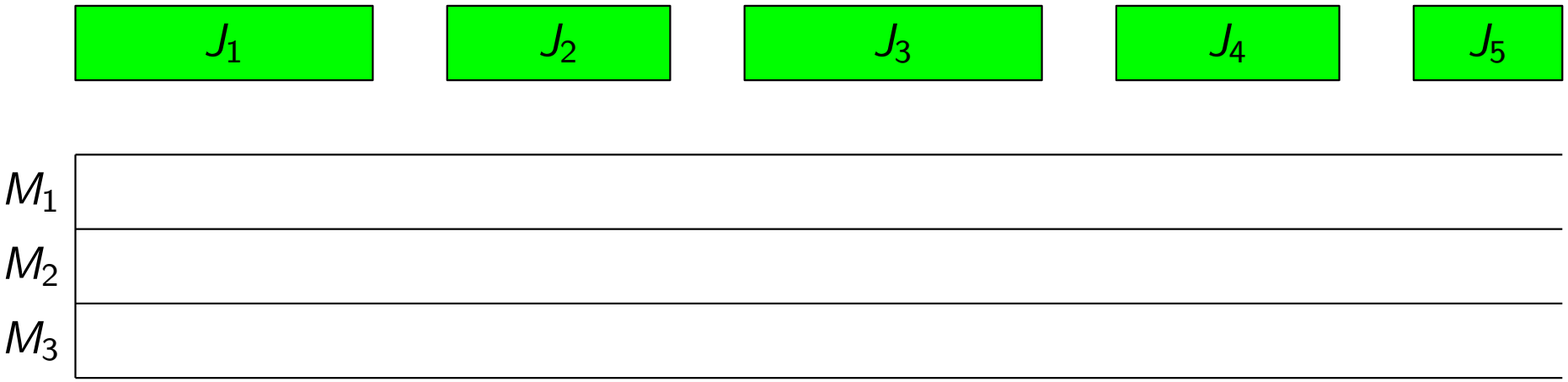
同一機械並列スケジューリング問題

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 $M_1, \ldots, M_m$，n 個のジョブ $J_1, \ldots, J_n$，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

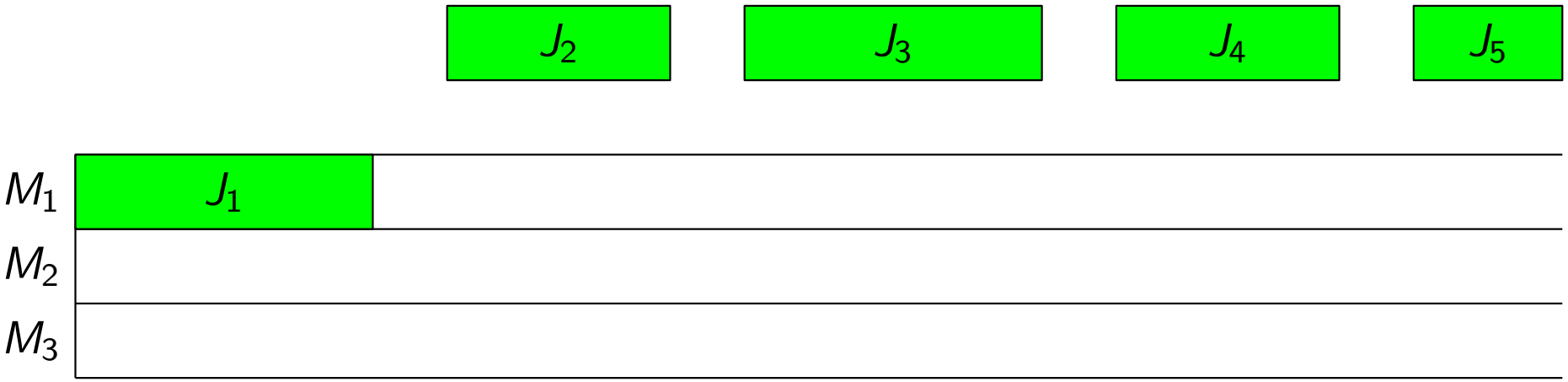
同一機械並列スケジューリング問題に対する貪欲法

1. $A_1, \ldots, A_m \leftarrow \emptyset$ とする (A_j が M_j で処理するジョブ集合となる)
2. $i = 1, \ldots, n$ 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 $A_1, \ldots, A_m$ を出力して停止．そうでなければ，2 に戻る



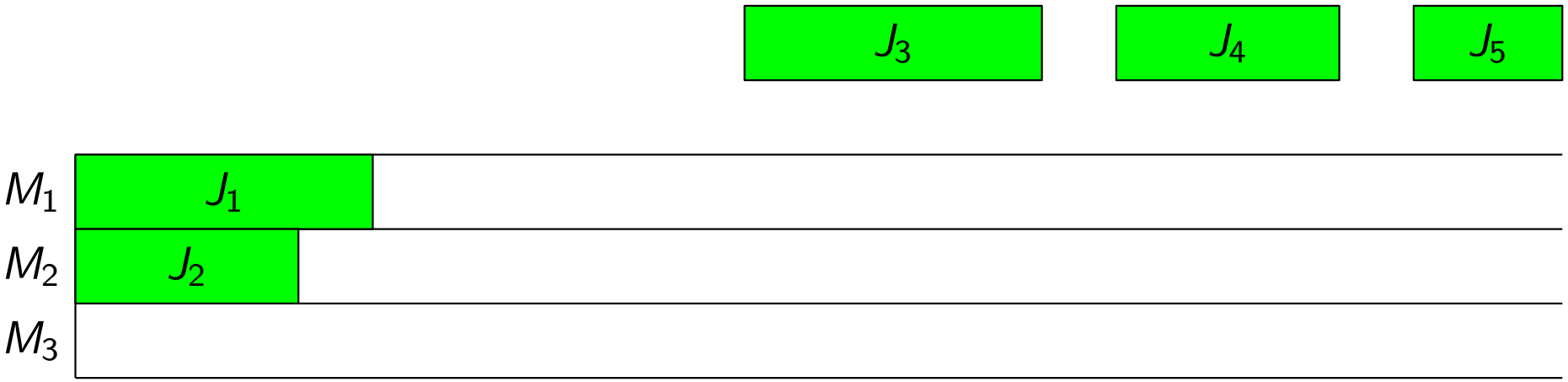
同一機械並列スケジューリング問題に対する貪欲法

1. $A_1, \ldots, A_m \leftarrow \emptyset$ とする (A_j が M_j で処理するジョブ集合となる)
2. $i = 1, \ldots, n$ 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 $A_1, \ldots, A_m$ を出力して停止．そうでなければ，2 に戻る



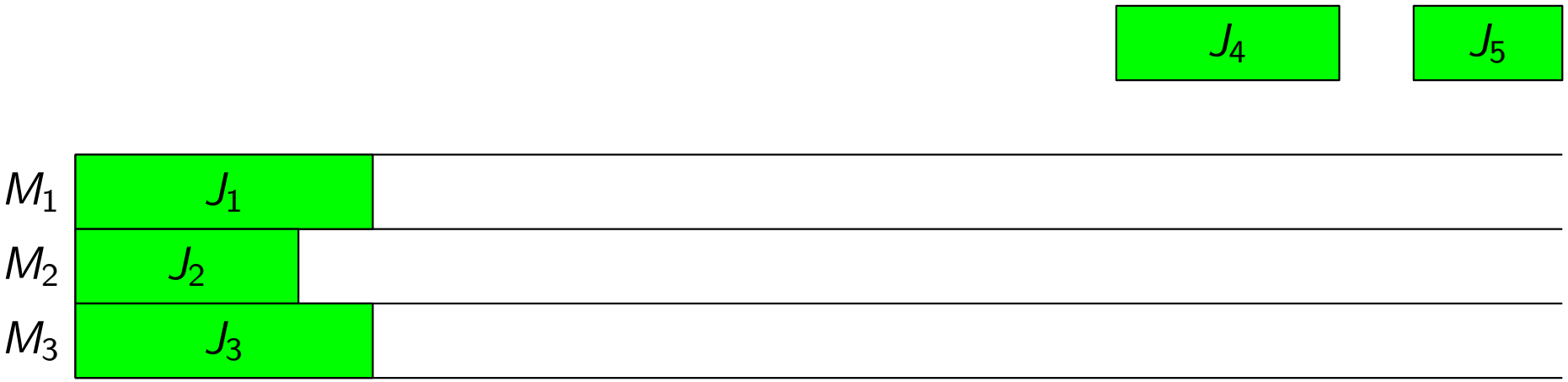
同一機械並列スケジューリング問題に対する貪欲法

1. $A_1, \ldots, A_m \leftarrow \emptyset$ とする (A_j が M_j で処理するジョブ集合となる)
2. $i = 1, \ldots, n$ 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 $A_1, \ldots, A_m$ を出力して停止．そうでなければ，2 に戻る



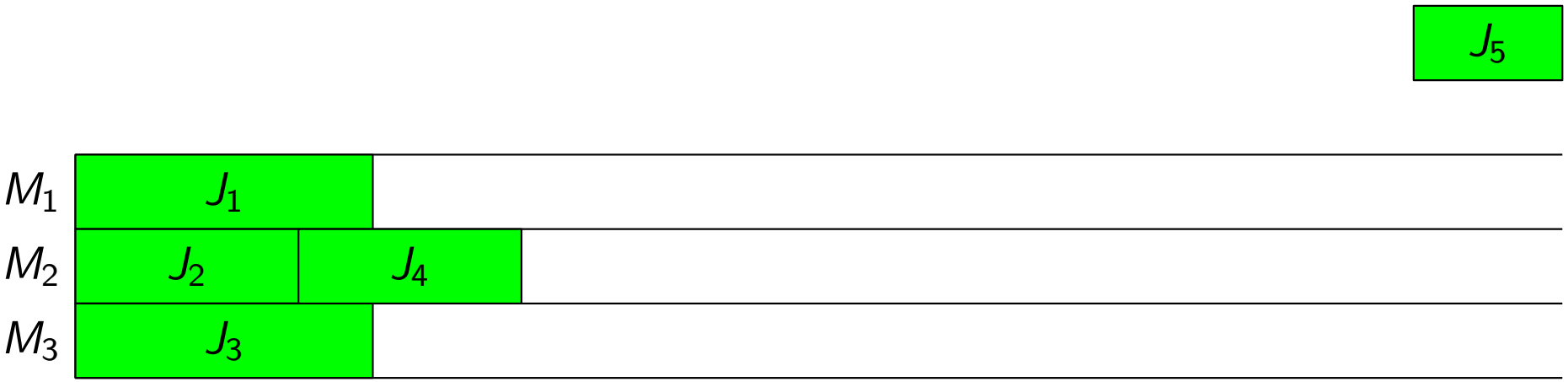
同一機械並列スケジューリング問題に対する貪欲法

1. $A_1, \ldots, A_m \leftarrow \emptyset$ とする (A_j が M_j で処理するジョブ集合となる)
2. $i = 1, \ldots, n$ 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 $A_1, \ldots, A_m$ 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. $A_1, \ldots, A_m \leftarrow \emptyset$ とする (A_j が M_j で処理するジョブ集合となる)
2. $i = 1, \ldots, n$ 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 $A_1, \ldots, A_m$ 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

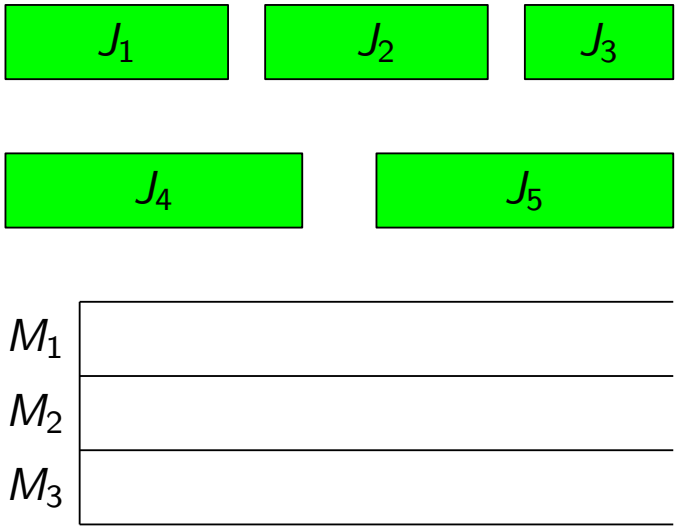
1. $A_1, \ldots, A_m \leftarrow \emptyset$ とする (A_j が M_j で処理するジョブ集合となる)
2. $i = 1, \ldots, n$ の順にジョブ J_i を見ていく
3. 機械の中で割り当てられたジョブの処理時間和が最小の機械 M_j を見つける (複数ある場合は，添字の最も小さいものとする)
4. $A_j \leftarrow A_j \cup \{J_i\}$ とする
5. $i = n$ ならば $A_1, \ldots, A_m$ を出力して停止．そうでなければ，2 に戻る

同一機械並列スケジューリングに対する貪欲法：山登り

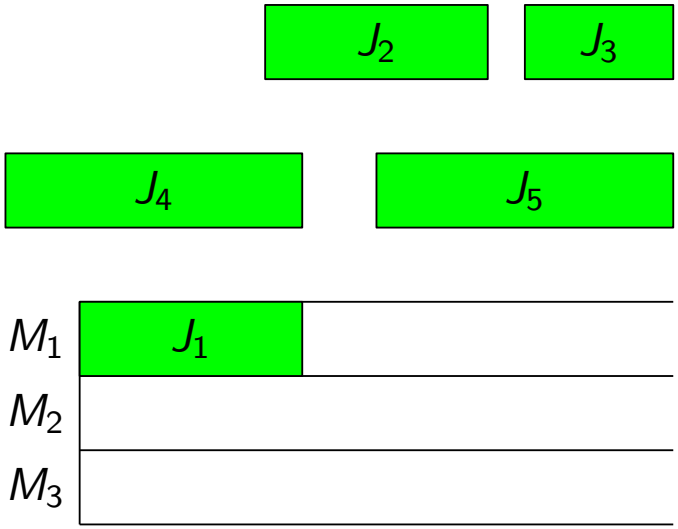
アルゴリズム設計で最初に確認するべきこと (再掲)

- 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？

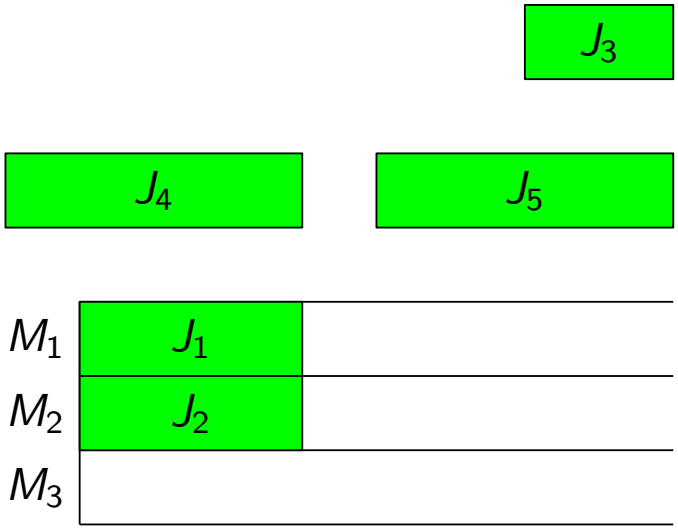
貪欲法が最適解を出力しない例



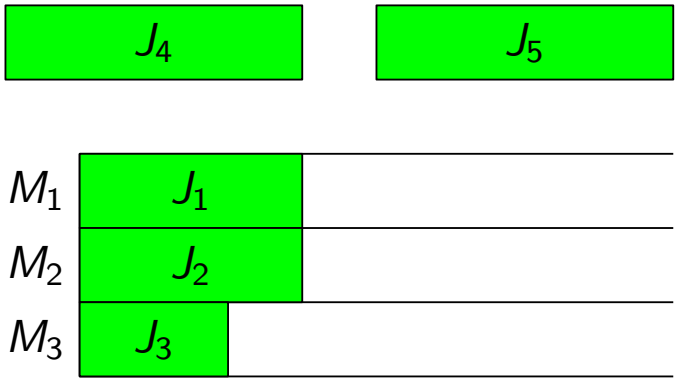
貪欲法が最適解を出力しない例



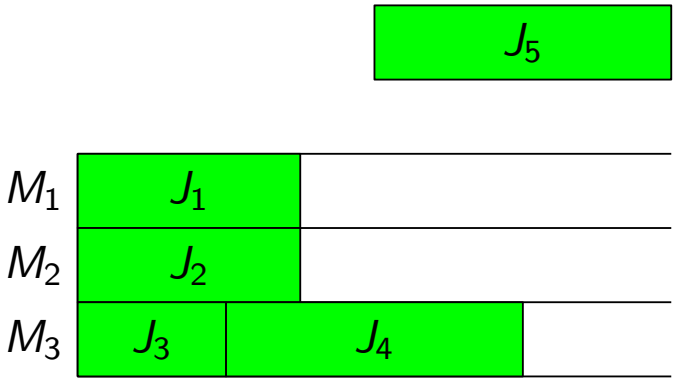
貪欲法が最適解を出力しない例



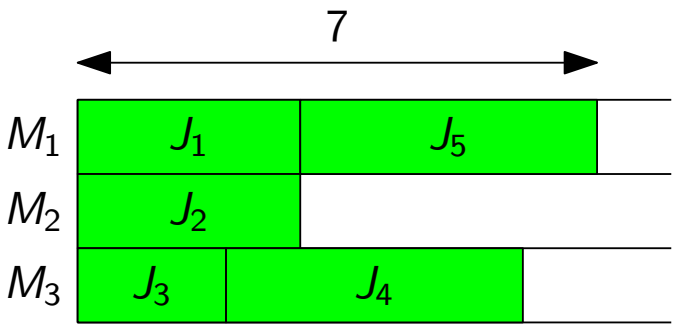
貪欲法が最適解を出力しない例



貪欲法が最適解を出力しない例

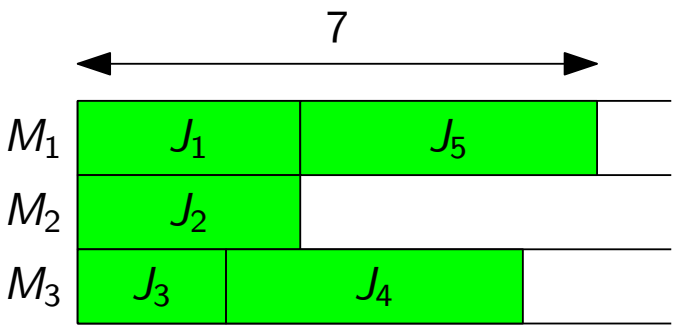


貪欲法が最適解を出力しない例

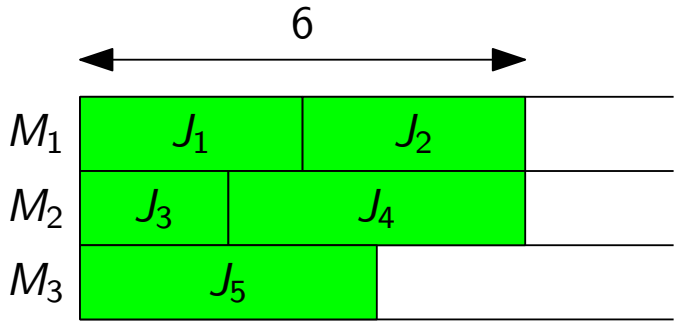


$\mathrm{ALG} = 7$

貪欲法が最適解を出力しない例



ALG $= 7$

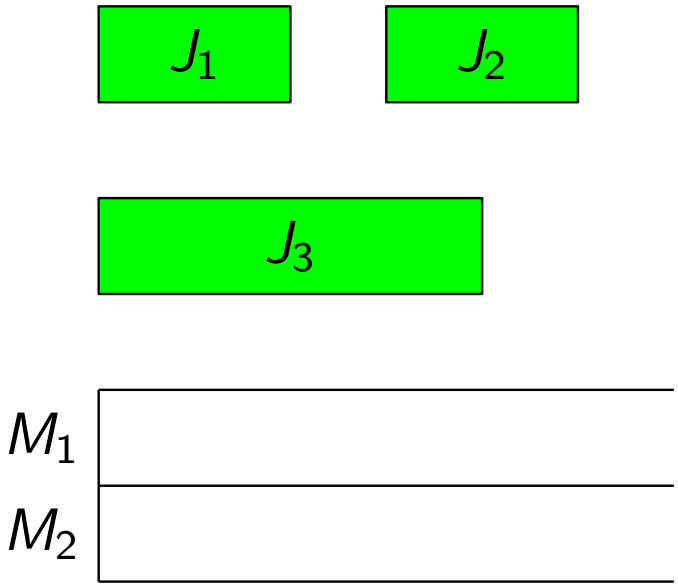


OPT ≤ 6

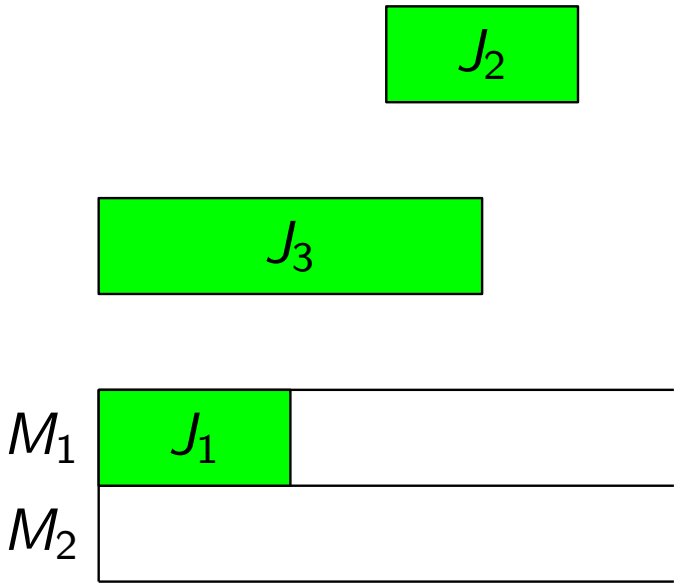
この例は

- **「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 $\quad (\because \text{ALG} = 7 > 6 \geq \text{OPT})$
- **「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 $\quad (\because \text{ALG} = 7 = \frac{7}{6} \cdot 6 \geq \frac{7}{6}\text{OPT})$

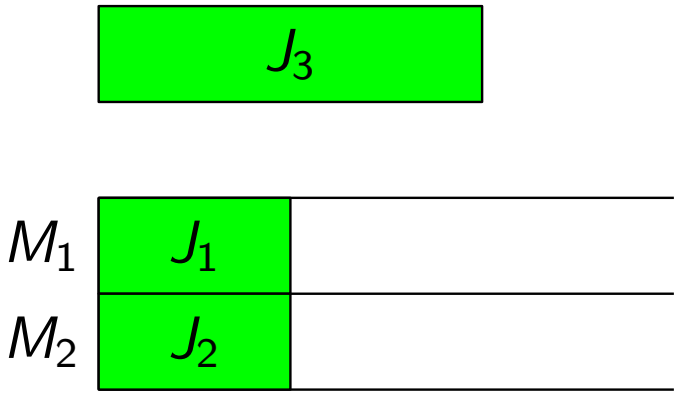
貪欲法の近似比が悪いことを示す例



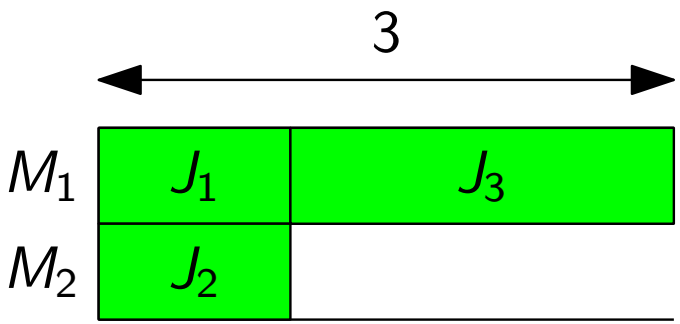
貪欲法の近似比が悪いことを示す例



貪欲法の近似比が悪いことを示す例

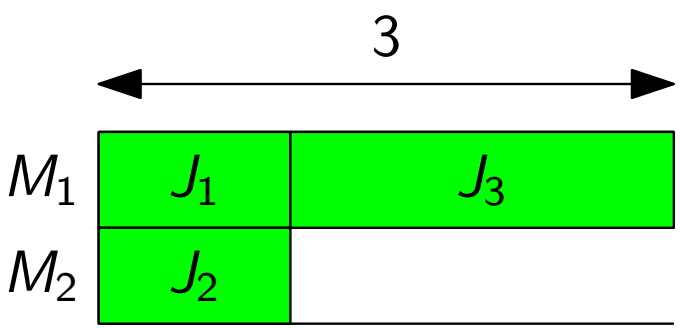


貪欲法の近似比が悪いことを示す例

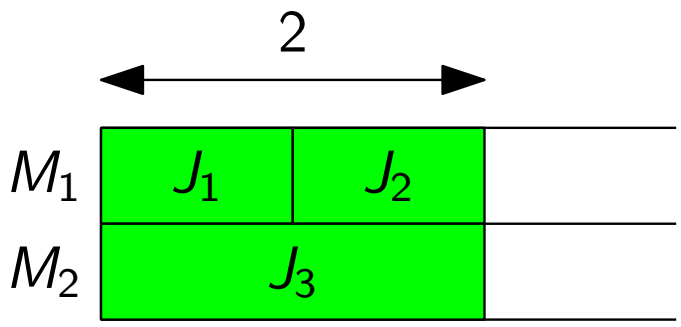


$\mathrm{ALG} = 3$

貪欲法の近似比が悪いことを示す例



ALG $= 3$



OPT ≤ 2

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**

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

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

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

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**)
- **∴ 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. $p_i \leq \mathrm{OPT}$
  2. $m \cdot \mathrm{OPT} \geq p_1 + p_2 + \cdots + p_n$
  3. $m \cdot (\mathrm{ALG} - p_i) \leq p_1 + p_2 + \cdots + p_{i-1} + p_{i+1} + \cdots + p_n$
- 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. $p_i \leq \mathrm{OPT}$

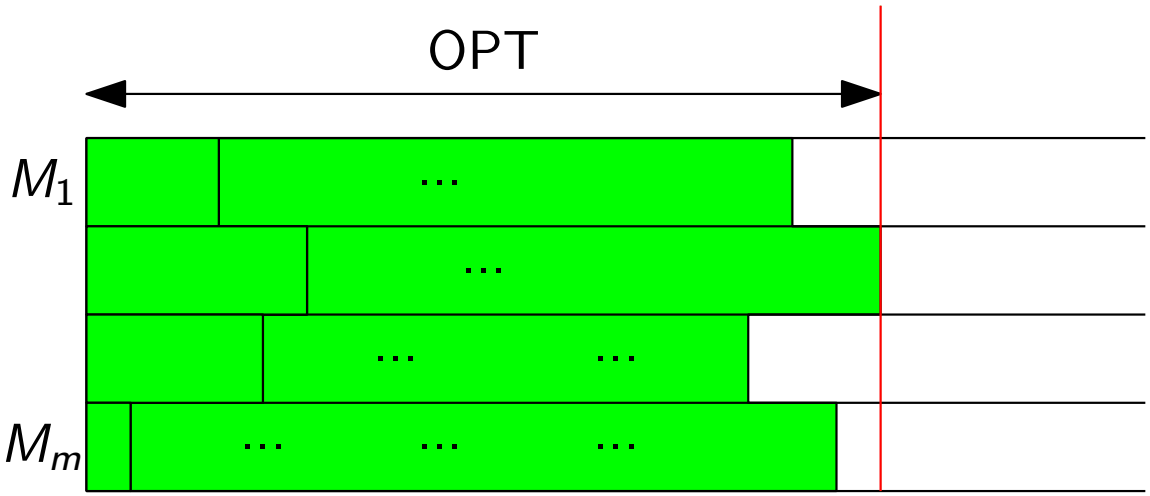
証明：これは簡単 (どうして？) □

貪欲法の近似比保証：証明 (2)

証明すべきこと

❷ $m \cdot \mathrm{OPT} \geq p_1 + p_2 + \cdots + p_n$

証明：OPT を与えている状況を表す次の図を見る □

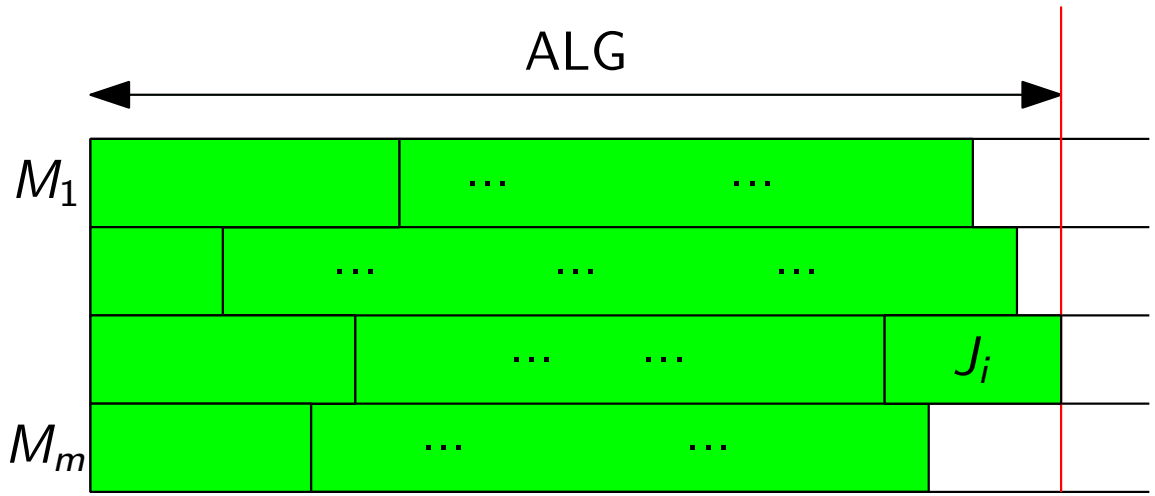


貪欲法の近似比保証：証明 (3)

証明すべきこと

❸ $m \cdot (\mathrm{ALG} - p_i) \leq p_1 + p_2 + \cdots + p_{i-1} + p_{i+1} + \cdots + p_n$

証明：ALG を与えている状況を表す次の図を見る



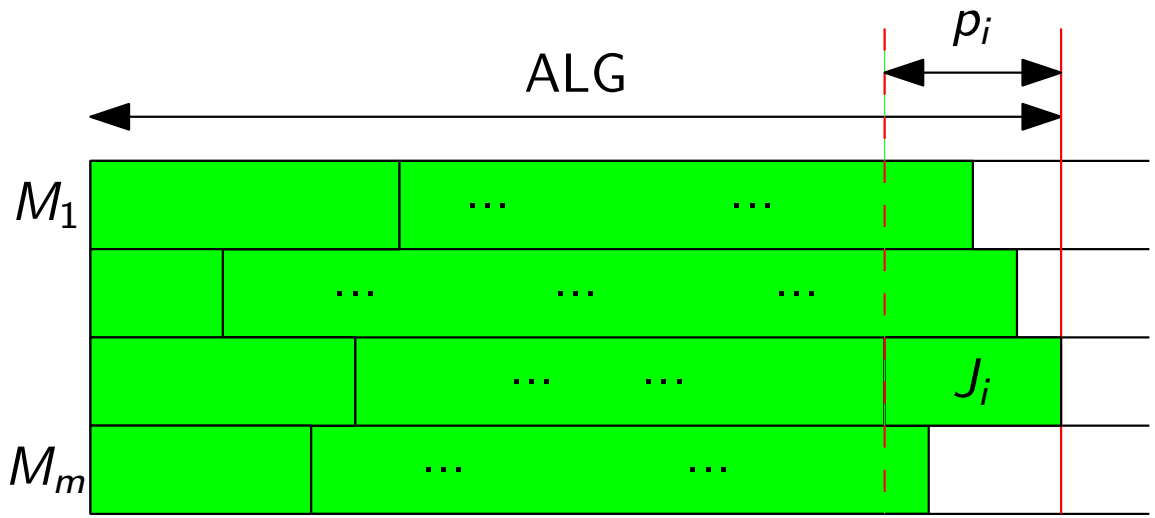
- J_i の処理の開始時刻に，どの機械も必ずジョブを処理している
  (どうして？)
- そこから，示したい式が得られる □

貪欲法の近似比保証：証明 (3)

証明すべきこと

❸ $m \cdot (\mathrm{ALG} - p_i) \leq p_1 + p_2 + \cdots + p_{i-1} + p_{i+1} + \cdots + p_n$

証明：ALG を与えている状況を表す次の図を見る



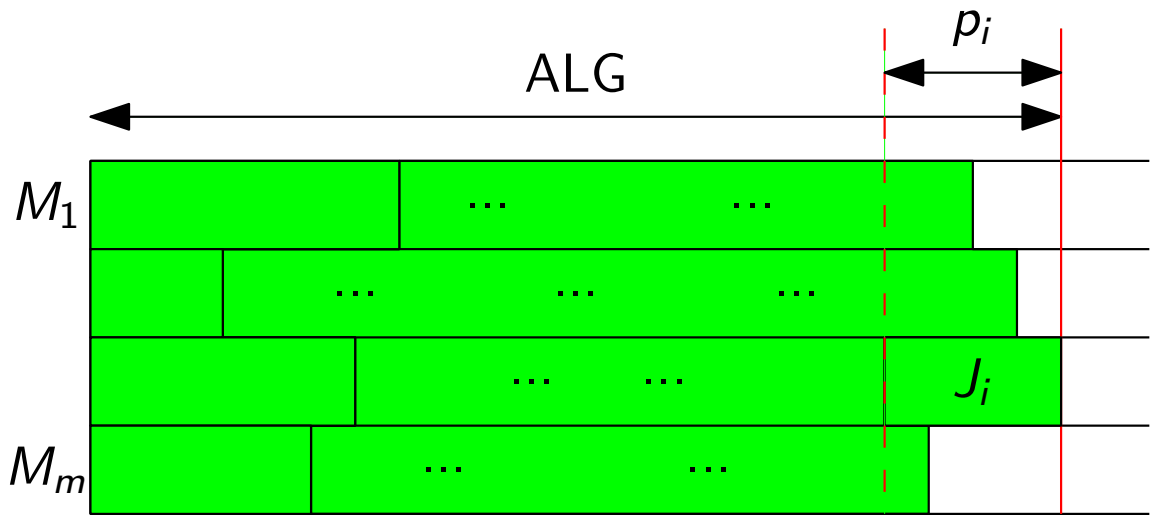
- J_i の処理の開始時刻に，どの機械も必ずジョブを処理している
  (どうして？)
- そこから，示したい式が得られる □

貪欲法の近似比保証：証明 (3)

証明すべきこと

❸ $m \cdot (\mathrm{ALG} - p_i) \leq p_1 + p_2 + \cdots + p_{i-1} + p_{i+1} + \cdots + p_n$

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 □

今日のまとめ

近似比 α

- $\mathrm{OPT} \leq \mathrm{ALG} \leq \alpha \cdot \mathrm{OPT}$ **(最小化問題)**
- $\mathrm{ALG} \leq \mathrm{OPT} \leq \alpha \cdot \mathrm{ALG}$ **(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るべきこと

- **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 2 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」，シュプリンガー・フェアラーク東京，**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

- **参照文献**

  - 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.