数理計画法 第6回
組合せ計画問題と分枝限定法

塩浦昭義
情報科学研究科 准教授

shioura@dais.is.tohoku.ac.jp
http://www.dais.is.tohoku.ac.jp/~shioura/teaching

中間試験について

- 日時：11月29日（木）13:00～14:30
- 11/22までにレポートを一度も出していない場合，受験不可
- 手書きのA4用紙一枚のみ持ち込み可（印刷やコピーは不可）
  - これも採点の対象，試験終了後に回収します
- 教科書，ノート等の持ち込みは不可
- 座席はこちらで指定
- 試験内容：11/15（第6回目，今日）までの講義で教えたところ
  - 様々な数理計画モデル
  - 線形計画問題の標準形，単体法，各種定理
  - 組合せ計画問題(分枝限定法)
- 50点満点，29点以下は原則として不合格

組合せ計画

分枝限定法を中心として

組合せ計画問題(組合せ最適化問題)

- 組合せ計画問題とは:
  - 有限個の「もの」の組合せの中から,
    目的関数を最小または最大にする組合せを見つける問題
  - 例1:整数計画問題全般(整数の組合せ)
  - 例2:グラフの最小木問題, 最短路問題, (グラフの枝の組合せ)
  - 例3:巡回セールスマン問題(都市の順列)

- 解きやすい問題と解きにくい問題
  - 解きやすい問題≒多項式時間で解ける問題
  - 解きにくい問題≒NP困難な問題
    (多項式時間で解けないと信じられている問題)

※注意:組合せ計画問題の解は有限個➔有限時間で必ず解ける!

生産計画問題の定式化

- 最大化：$70x_1 + 120x_2 + 30x_3$
- 条件：

$$
\begin{aligned}
5x_1 \qquad\quad + 6x_3 &\le 80 \\
2x_2 + 8x_3 &\le 50 \\
7x_1 \qquad\quad + 15x_3 &\le 100 \\
3x_1 + 11x_2 \qquad\quad &\le 70 \\
x_1 \ge 0, x_2 \ge 0, x_3 &\ge 0
\end{aligned}
$$

x_1, x_2, x_3 は整数

変数（の一部）に
整数条件が付加
➔整数計画問題

整数計画問題の例2: ナップサック問題

- ハイキングの準備
- n個の品物の中から持って行くものを選択
- ナップサックには b kg まで入れられる
- 品物 $i = 1,2,\ldots,n$ の重さは a_i kg, 利用価値は c_i
- 利用価値の合計を最大にしたい

↓

最大化: $\sum_{i=1}^{n} c_i\, x_i$

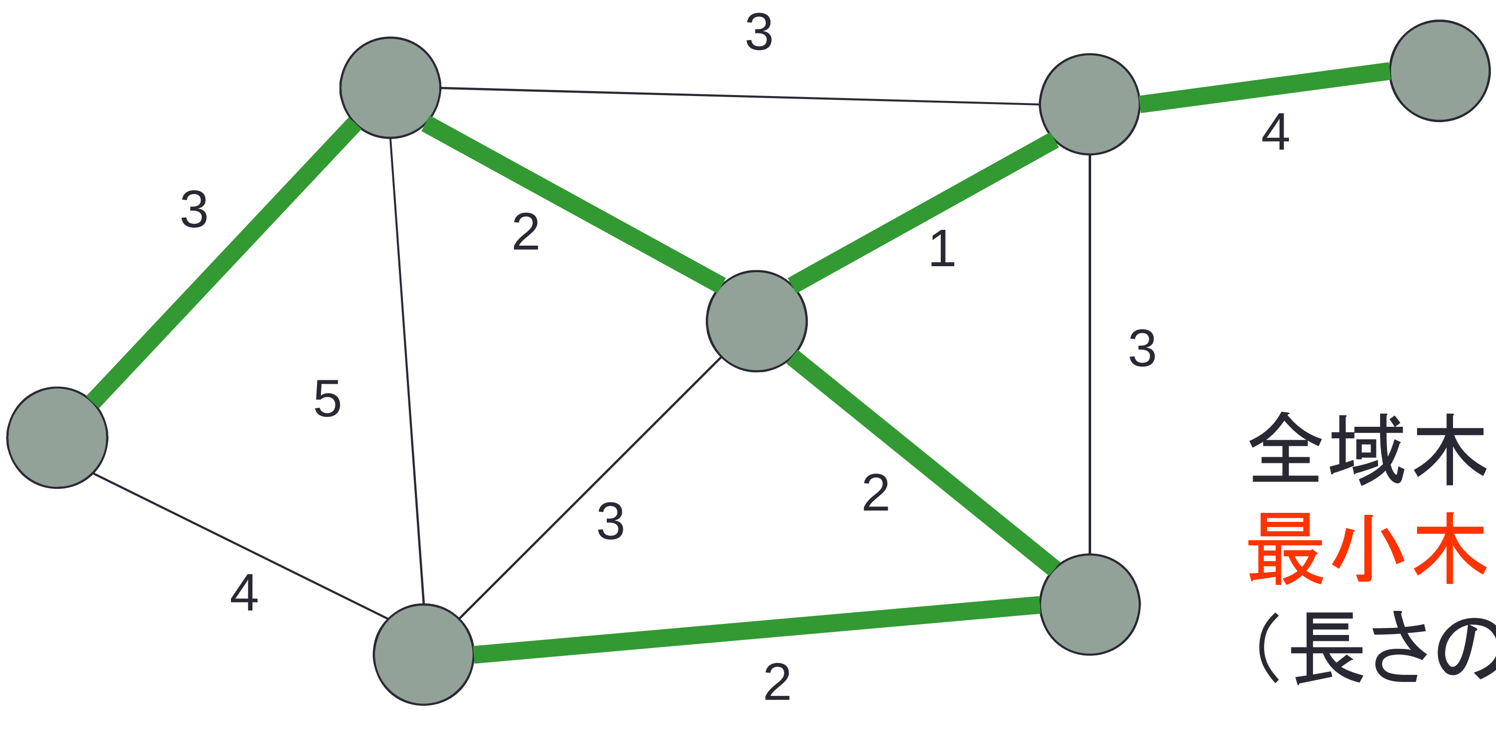
条件: $\sum_{i=1}^{n} a_i\, x_i \le b$

$x_1, x_2, \ldots, x_n \in \{0,1\}$

変数の全てが
0または1
➔0-1整数計画問題

最小木問題

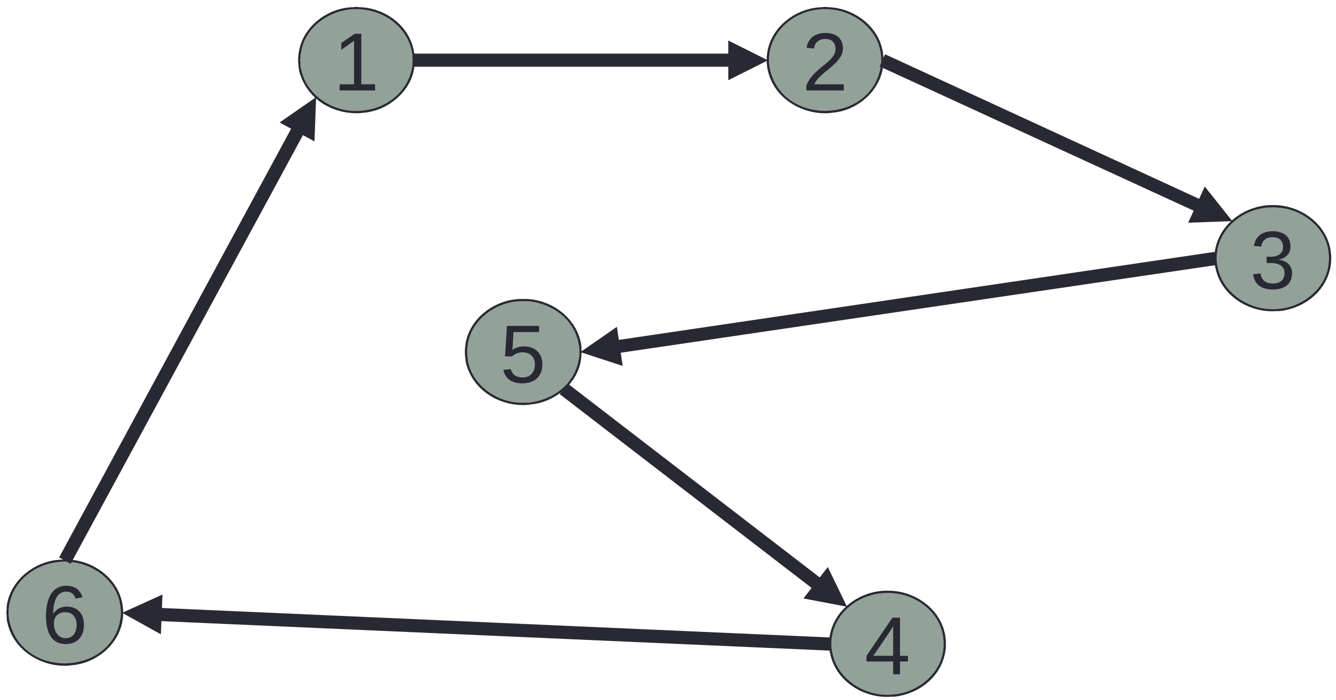
- 入力：無向グラフG=(V,E), 各枝の長さd(e) (e∈E)
- 出力：Gの最小木（Gの全域木で，枝の長さの和が最小のもの）



全域木であり，
最小木である
（長さの和＝14）

巡回セールスマン問題

- セールスマンが n 都市をちょうど一回ずつ巡回する
- 都市 i から j への距離は c_{ij}（平面上の距離で与えられるケースも多い）
- 目的：都市を巡回する際の総距離を最小にする



組合せ計画問題に対するアプローチ

- 組合せ計画問題をどのように解くか？
- 解きやすい問題の場合
  - 多項式時間アルゴリズムを構築➔より高速な解法へ
- 解きにくい問題の場合
  - 絶対に最適解が必要な場合➔厳密解法
    - 分枝限定法（授業で説明）🡨現在の主流
    - 動的計画法（「アルゴリズムとデータ構造」の講義）
  - ある程度良い解であれば十分という場合
    - 精度保証付き近似アルゴリズム
      （解の良さに対する理論保証あり）
    - ヒューリスティックス（解の良さは実験的に証明）

組合せ計画問題に対する厳密解法

- 組合せ計画問題は解を全列挙すれば解ける
- しかし，計算時間が膨大で現実には不可能

  ➔ 解の全列挙における無駄を出来るだけ省く

  - 動的計画法：同一の部分問題を繰り返し解くことを避ける
  - 分枝限定法：ある部分問題から最適解が得られないことがわかったら，その部分問題は無視する

分枝限定法の考え方

- 組合せ計画問題を，場合分けによって部分問題に分解（分枝操作）
  - 0–1ナップサック問題：各変数について 0 の場合と 1 の場合に分ける
- 分枝の進行の様子は探索木により表現可能
- これだけでは，解の全列挙と同じで時間がかかる

0-1ナップサック問題の探索木

最大化 $10x_1 + 7x_2 + 25x_3$

条件 $2x_1 + x_2 + 6x_3 \le 7$

$x_1, x_2, x_3 \in \{0,1\}$

x_1を0に
固定

最大化 $7x_2 + 25x_3$

条件 $x_2 + 6x_3 \le 7$

$x_2, x_3 \in \{0,1\}$

部分
問題

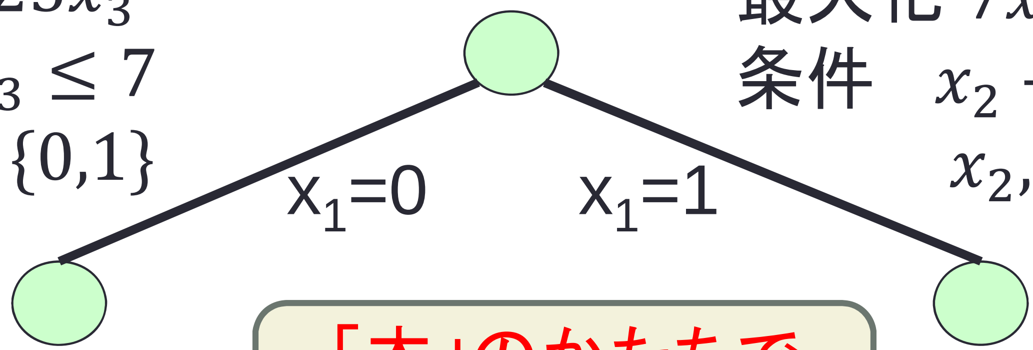
x_1を1に
固定

最大化 $7x_2 + 25x_3 + 10$

条件 $x_2 + 6x_3 \le 5$

$x_2, x_3 \in \{0,1\}$

部分
問題



「木」のかたちで
表現する

0-1ナップサック問題の探索木

最大化 $10x_1 + 7x_2 + 25x_3$

条件 … 7

x_1を0に固定

x_1を1に固定

さらに細かく
場合分けする

最大化 $7x_2 + 25x_3$

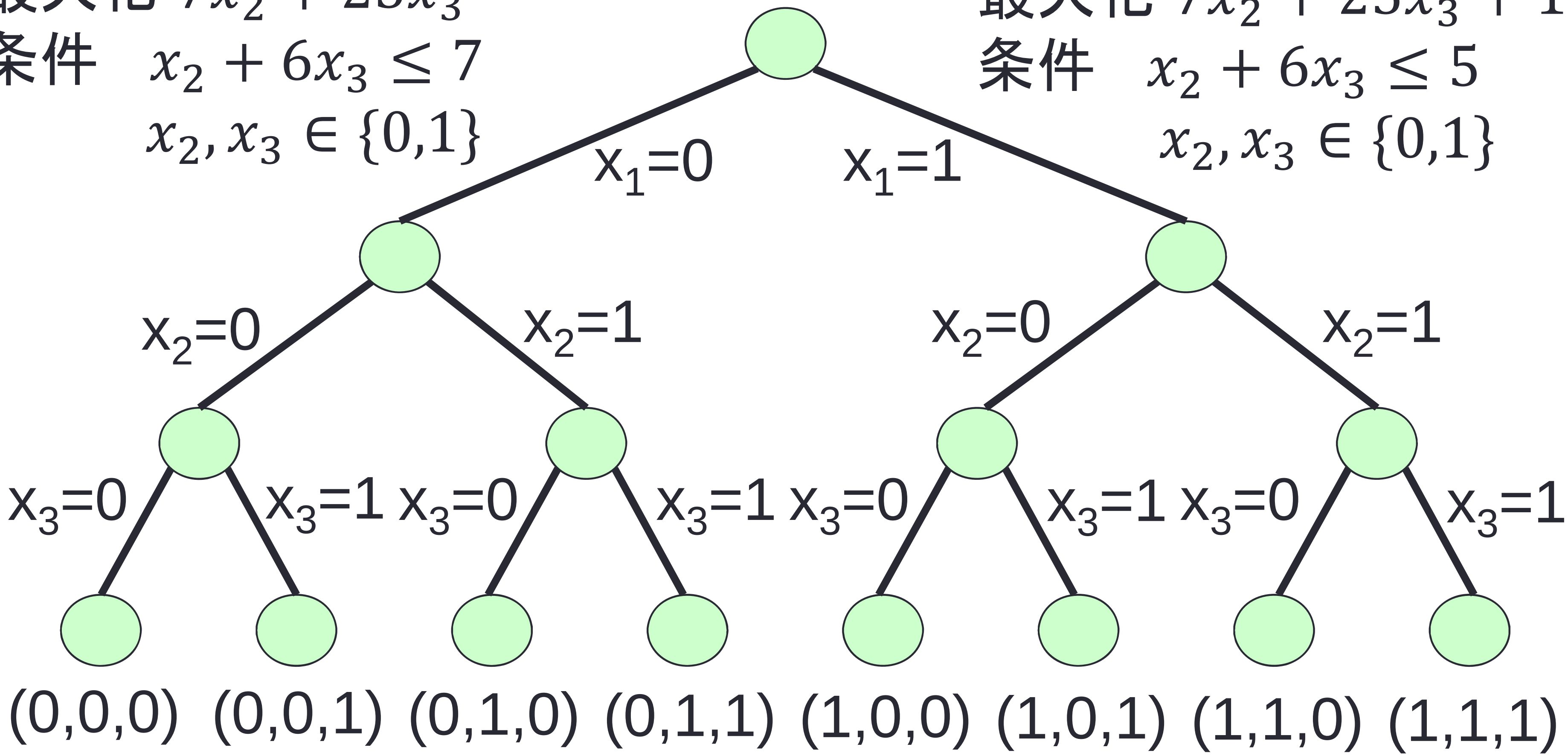
条件 $x_2 + 6x_3 \le 7$

$x_2, x_3 \in \{0,1\}$

最大化 $7x_2 + 25x_3 + 10$

条件 $x_2 + 6x_3 \le 5$

$x_2, x_3 \in \{0,1\}$



分枝限定法の考え方

- 分枝操作により，たくさんの部分問題が生成される
- 解く必要のない（解いても無駄な）部分問題が検出されたら，
  さらなる分枝操作をストップ（限定操作）

わかりやすい例：

最大化 $100x_1 + 7x_2 + 25x_3$
条件 $x_1 + x_2 + 6x_3 \leq 7$
$x_1, x_2, x_3 \in \{0,1\}$

品物1は価値が十分に大きく，重さが小さい
➔最適解は品物1を必ず含む
➔$x_1 = 0$ の場合は考える必要なし（限定操作）

分枝限定法の考え方

- 分枝操作により，たくさんの部分問題が生成される
- 解く必要のない（解いても無駄な）部分問題が検出されたら，

  さらなる分枝操作をストップ（限定操作）

判断が難しい例：

$$\begin{aligned} &\text{最大化}\ 32x_1 + 14x_2 + 30x_3 \\ &\text{条件}\quad 6x_1 + 2x_2 + 5x_3 \le 7 \\ &\qquad\quad x_1, x_2, x_3 \in \{0,1\} \end{aligned}$$

限定操作を行うための
上手なチェック方法は？

品物1は価値も重さも大きい
➔最適解は品物1を必ず含むかどうか，わからない
➔$x_1 = 0$ の場合と$x_1 = 1$の場合の両方を考える必要があるかもしれない

分枝限定法の考え方

- 限定操作を行うための上手なチェック方法は？
- チェック手段その1：暫定解をつねに保持する
  - 暫定解：分枝限定法の現時点までの計算により
    得られた最良の許容解
- 解く必要のない部分問題の例
  - （部分問題の）最適解がすでに得られた
  - （部分問題に）許容解が存在しない
  - 現在の暫定解より良い許容解を得られる可能性がない

どうやって判定する？

- チェック手段その2：緩和問題を使って最適値の上界値を計算
  （最大化の場合）

緩和問題

- 緩和問題：元の数理計画問題の
  制約条件の一部を緩和して得られる問題

目的関数：$\sum_{i=1}^{n} c_i\, x_i \rightarrow$ 最大
制約条件：$\sum_{i=1}^{n} a_i\, x_i \leq b$
$x_1, x_2, \ldots, x_n \in \{0,1\}$

0-1ナップサック問題
（解きにくい）

$x_j \in \{0,1\}$を$0 \leq x_j \leq 1$に緩和

目的関数：$\sum_{i=1}^{n} c_i\, x_i \rightarrow$ 最大
制約条件：$\sum_{i=1}^{n} a_i\, x_i \leq b$
$0 \leq x_j \leq 1\ (j = 1,2, \ldots, n)$

連続ナップサック問題
（解きやすい）

連続ナップサック問題は線形計画問題➔多項式時間で解ける
O(n)時間アルゴリズムが存在（「アルゴリズムとデータ構造」）

緩和問題の性質

- 緩和問題は元の問題より解きやすい（簡単に解ける）ことが多い
  - 通常，簡単に解ける問題を緩和問題として選ぶ
- 緩和問題は元の問題の条件を緩和した問題

  ➔緩和問題の実行可能解集合は，元の問題の
  実行可能解集合を含む

  ➔緩和問題の最大値≧元の問題の最大値

  ∴緩和問題の最適値（最大値）は，元の問題の最適値の上界
- 緩和問題の最適解を修正することにより，元の問題の実行可能解を作ることが可能なケースが多い
  - 緩和問題の最適解が元の問題の実行可能解

    ➔元の問題の最適解になっている！

0-1ナップサック問題の緩和問題：例

緩和問題の最適解は，c_j/a_j の大きい方から順に変数を増やしていけば得られる

	1	2	3	4	5	6	7	8
c_j	15	100	90	60	40	15	10	1
a_j	2	20	20	30	40	30	60	10

b=102

a_jの合計72≦b

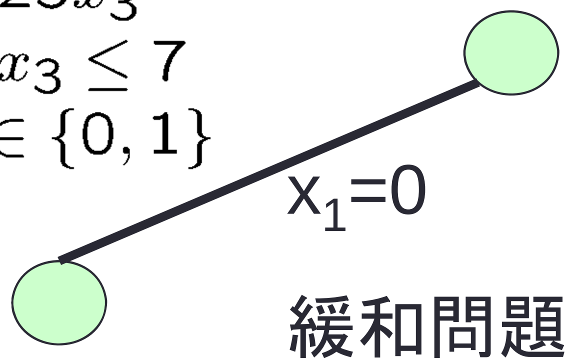
(1,1,1,1,(102-72)/40,0,0,0)は最適解
目的関数値＝295
≧0-1ナップサック問題の最適値

$x_5 = 0$に置き換えた解(1,1,1,1,0,0,0,0)は
元問題の実行可能解，目的関数値＝265

解く必要のない部分問題の例：最適解が得られた部分問題

$$
\begin{array}{ll}
\text{最大化} & 10x_1 + 7x_2 + 25x_3 \\
\text{条件} & 2x_1 + x_2 + 6x_3 \leq 7 \\
 & x_j \in \{0,1\} \ (j = 1,2,3)
\end{array}
$$

$$
\begin{array}{ll}
\text{最大化} & 7x_2 + 25x_3 \\
\text{条件} & x_2 + 6x_3 \leq 7 \\
 & x_2, x_3 \in \{0,1\}
\end{array}
$$



緩和

$$
\begin{array}{ll}
\text{最大化} & 7x_2 + 25x_3 \\
\text{条件} & x_2 + 6x_3 \leq 7 \\
 & 0 \leq x_2, x_3 \leq 1
\end{array}
$$

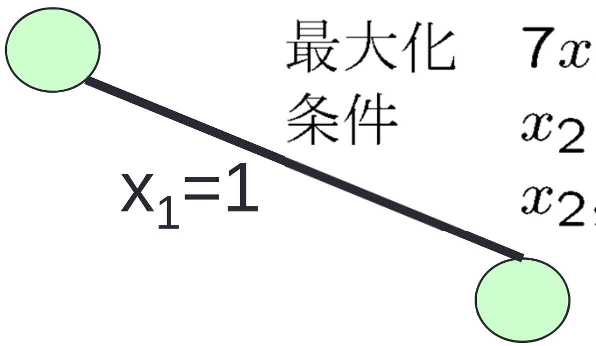
緩和問題の最適解は
$(x_2, x_3) = (1, 1)$（整数解）
➔元の**部分問題の最適解**

$(x_1,x_2,x_3) = (0,1,1)$ は暫定解，
目的関数値＝32

この部分問題をさらに調べても，
より良い解は得られない
➔この部分問題の**探索をストップ**

解く必要のない部分問題の例：より良い解が得られない部分問題

$(x_1, x_2, x_3) = (0,1,1)$
は暫定解,
目的関数値＝32



最大化　$7x_2 + 25x_3$

条件　$x_2 + 6x_3 \leq 5$

$x_2, x_3 \in \{0, 1\}$

緩和問題の最適解は
$(x_2, x_3) = (1, 2/3)$
目的関数値＝23.6666…
これは部分問題の上界値,
≦ 暫定解の目的関数値32

緩和

最大化　$7x_2 + 25x_3$

条件　$x_2 + 6x_3 \leq 5$

$0 \leq x_2, x_3 \leq 1$

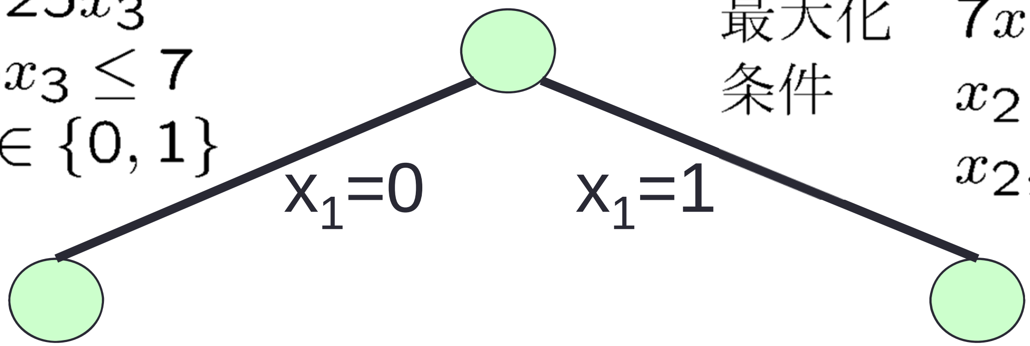
この部分問題をさらに調べても,
暫定解より良い解は得られない
➔この部分問題の探索をストップ

解く必要のない部分問題の例：実行不可能な部分問題

$$
\begin{array}{ll}
\text{最大化} & 10x_1 + 7x_2 + 25x_3 \\
\text{条件} & \boxed{8x_1} + x_2 + 6x_3 \leq 7 \\
 & x_1, x_2, x_3 \in \{0, 1\}
\end{array}
$$

$$
\begin{array}{ll}
\text{最大化} & 7x_2 + 25x_3 \\
\text{条件} & x_2 + 6x_3 \leq 7 \\
 & x_2, x_3 \in \{0, 1\}
\end{array}
$$

$$
\begin{array}{ll}
\text{最大化} & 7x_2 + 25x_3 \\
\text{条件} & x_2 + 6x_3 \leq -1 \\
 & x_2, x_3 \in \{0, 1\}
\end{array}
$$



この部分問題は実行不可能

➔この部分問題の探索をストップ

分枝限定法の流れ

記号 L: 部分問題のリスト,
x^*: 暫定解, z: 暫定値(暫定解の目的関数値)
ステップ0:L={元問題}, z=−∞, x^*は未定義とする.
ステップ1(探索):
Lが空ならば計算終了. 現在の x^* が最適解.
Lが非空ならば, Lから部分問題P'を選び, 削除.
ステップ2(限定操作):
2−a: P' が実行不可能であることがわかったら, ステップ1へ.
2−b: P'の最適解が得られたら, 必要に応じて x^*, z を更新してステップ1へ.
2−c: P' の緩和問題を解いた結果, 暫定解より良い許容解が得られないことがわかったらステップ1へ.

分枝限定法の流れ

ステップ2（限定操作）:

2−a: P’ が実行不可能であることがわかったら，ステップ1へ.

2−b: P’の最適解が得られたら，必要に応じて x^*, z を更新してステップ1へ.

2−c: P’ の緩和問題を解いた結果，暫定解より良い許容解が得られないことがわかったらステップ1へ.

ステップ3（分枝操作）:

P’を場合分けによってP'_1, P'_2, …, P'_kに分解.

L にこれらの問題を入れ，ステップ1へ.

分枝限定法の実装における検討事項

分枝限定法の性能は，各々のステップを
如何に実現するかによって左右される

どのような順番で
部分問題を選ぶか？
（部分問題の探索法）

ステップ1（探索）：
Lが空ならば計算終了．現在の x^* が最適解．
Lが非空ならば，Lから部分問題P’を選び，削除．

ステップ2（限定操作）：
2−a: P’ が実行不可能であることがわか
2−b: P’の最適解が得られたら，必要に
ステップ1へ．
2−c: P’ の緩和問題を解いた結果，暫定
得られないことがわかったらステップ

どのように実行不可能
性を判定するか？

どのような緩和問題を
解くか？

ステップ3（分枝操作）：
P’を場合分けによってP’$_1$, P’$_2$, …, P’$_k$に分解．
L にこれらの問題を入れ，ステップ1へ．

どのように問題を
分解するか？

分枝限定法の実装における検討事項

- 部分問題の探索法
  - どのような順番で部分問題を選ぶか？
- 限定操作のやり方
  - どのように実行不可能性を判定するか？
  - どのような緩和問題を解くか？
- 分枝操作のやり方
  - どのように問題を分解するか？

レポート問題

下記のナップサック問題を分枝限定法で解きなさい。
ただし，「ナップサックの容量＝10」とする
注意事項：

- 変数をx_A, x_B, x_C, x_Dとする
- x_A, x_B, x_C, x_Dの順に変数を0または1に固定して部分問題を生成する
- 変数を0に固定した問題を優先して解く

品物	A	B	C	D
価値	25	9	11	17
重さ	8	3	4	7