●最適化問題の解法について（復習）

連続変数を対象とする最適化（Linear Programming）

生産計画問題
輸送問題
栄養問題

$X_{ij} \geqq 0$：*非負条件*

対処方法

列挙法　（シンプレクス法，双対シンプレクス法）

整数変数を対象とする最適化（組合せ最適化）

ナップサック問題
割当て問題
巡回セールスマン問題

$X_{ij} \in \{0,1\}$

対処方法

列挙法　（分枝限定法，整数計画法，総当たり法）
発見的方法（Greedy Method, Stingy Method）
探索法　（局所探索法，遺伝的アルゴリズム）

●分枝限定法(Branch and Bound Method)

・組合せ最適化問題の有力な解法

組合せ最適化問題・・・・原理的に総当たり法で解ける。
→計算爆発する可能性あり。

分枝限定法・・・・実行可能解を列挙＆場合分けする。
→少ない計算回数で解を求める。

原理

解が得られる可能性が無い部分を計算しないことで、総計算回数を削減する。

●ナップサック問題

いくつかの荷物と一定容量まで荷物を入れられる袋がある．各荷物の容量と価格が定められているとき，荷物の総価格が最大になるように袋へ詰めるにはどうすれば良いか？

＜定式化の考え方＞

袋の容量をＢ．
全部でＮ個の荷物．
荷物ｉの容量を Ai，価格を Ci
荷物ｉを袋に入れることを Xi=1，
荷物ｉを袋に入れないことを Xi=0

<定式化>通常のナップサック問題

$$\max \sum_{i=1}^{N} C_i \cdot X_i$$

$$\text{subject to} \quad \sum_{i=1}^{N} A_i \cdot X_i \leqq B,\ X_i \in \{0,1\} \quad (i=1,\cdots,N)$$

<緩和問題>分枝限定法で解く。

$$\max \sum_{i=1}^{N} C_i \cdot X_i$$

$$\text{subject to} \quad \sum_{i=1}^{N} A_i \cdot X_i \leqq B,\ 0 \leqq X_i \leqq 1 \quad (i=1,\cdots,N)$$

ただし$(C_1/A_1 \geqq C_2/A_2 \geqq C_3/A_3 \geqq \cdots \geqq C_n/A_n)$とする。

●分枝限定法の例題

Max　$5X_1 + 6X_2 + X_3 + 2X_4$

Subject to

$4X_1 + 5X_2 + X_3 + 3X_4 \leqq 6$　　　$X_i = (0, 1)$　$i=1,2,\cdots,4$

↓

<u>緩和問題へ変換</u>

Max　$5X_1 + 6X_2 + X_3 + 2X_4$

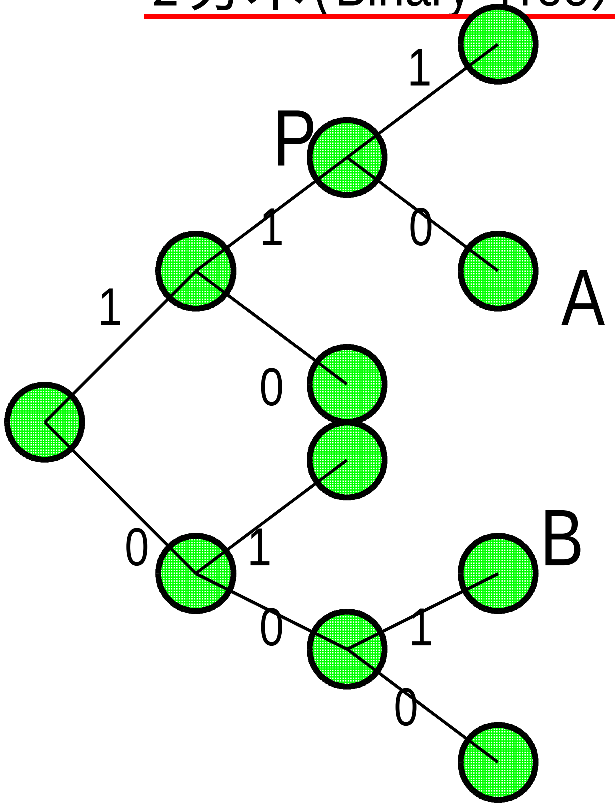
Subject to

$4X_1 + 5X_2 + X_3 + 3X_4 \leqq 6$　　$0 \leqq X_i \leqq 1$　　$i=1,2,\cdots,4$

★制約条件を満たす X_i を（$0 \leqq X_i \leqq 1$）の範囲で左側から定めていく。まず$(X_1,X_2,X_3,X_4)=(1,0,0,0)$とすると、$4 \leqq 6$ が成立。まだ OK なので X_2 を 2/5 とおき(1,2/5,0,0)とすると、制約式は $6 \leqq 6$ となる。従って(1,2/5,0,0)は緩和問題の実数最適解となる。

●組合せ最適化問題（0－1 最適化問題）について

→ 2分木(Binary Tree）によって解空間を表現可能 → 分岐図



A=(1 1 0), B=(0 0 1)で表す。

P は P=(1 1)であるが、P 以降の節点は、P(1 1 ?)と表現可能。すなわち A を含む。

部分問題：P(J_0, J_1)

J_0：0に固定された添字の集合

J_1：1に固定された添字の集合

F：固定されていない添字の集合

（例）

F={1,3}, J0={2}, J1={4}に対する部分問題 P({2},{4})は

Max $5X_1 + X_3 + 2$

Subject to

$4X_1 + X_3 + 3 \leqq 6$ $X_i=(0, 1)$ $i=\{1, 3\}$

●分子限定法の手順

1）原問題から緩和問題を作る。
2）緩和問題から実数最適解を得る。
3）実数最適解を改良して近似最適解（暫定解）を得る。
4）分枝操作によって部分問題を作る。
5）部分問題の最適解を求め、それが原問題の最適解となる
可能性があるかどうかをチェックする。
6）もし可能性がないならその部分問題を解くことを止める（限定操作）
7）新たに、次の部分問題を作る（分枝操作）
8）新しい部分問題をチェックする。
9）チェックすべき部分問題が無くなれば終了。