10　いくつかの補足

”Combinatorial Optimization”を読む上で知っておいた方が良いが教科書では省略されている内容をこの章に記す．

10.1　ヒープ

ヒープは木を使ったデータ構造の一種であり，最小値（もしくは最大値）を高速に求められるという特徴がある．ヒープにもいくつかの種類があるが，ここでは二分ヒープについて説明する．

定義 10.1 二分木（binary tree）
グラフ理論において，根付き木であって各頂点について子が 2 個以下であるようなものを根付き二分木，あるいは単に二分木という．根付き二分木の各頂点にデータを保存するデータ構造のことも二分木という．

二分木は次の図のように，各頂点になんらかのデータと親と左右の子へのポインタをもたせることで実装される．

定義 10.2 二分ヒープ（binary heap）
二分木のデータ構造であって，次の 2 つの条件をみたすようにしたものを最大二分ヒープという．
(1) 各頂点に保存されている値は子の値以上である．
(2) 各頂点は高さが低い順に，同じ高さでは左から順にデータが埋まっている．
(1) の条件を *max-heap property*，(2) の条件を *shape property* という．(1) の「以上」の部分を「以下」にしたものを最小二分ヒープという．

二分ヒープのことを単にヒープということもある．二分ヒープは親と子の添字の計算が簡単にできるため，配列を用いて実装することが可能である．実際に，高さが低い順に，同じ高さでは左から順に 1 から添字付けすると，次のように添字を返す関数がかける．

```
1: function PARENT(i)                          ▷ 親の添字を返す関数
2:     return ⌊i/2⌋
3: function LEFT(i)                            ▷ 左の子の添字を返す関数
4:     return 2i
5: function RIGHT(i)
6:     return 2i + 1                           ▷ 右の子の添字を返す関数
```

次の主張は，二分ヒープの高さが $\Theta(\log n)$ であることによって成立する．

主張 10.1
最大二分ヒープにおいて，最大値の取得の計算量は $O(1)$，要素の追加・削除の計算量は $O(\log n)$ でできる．

また，未整列のリストの要素をすべてヒープに追加して，その後すべて取り出すことで，整列したリストを得ることができる．このソートの方法をヒープソートという．

主張 10.2
ヒープソートの計算量は $O(n \log n)$.

10.2　グラフのデータ構造

グラフを表現する方法として，接続行列や隣接行列，隣接リストが知られている．各表現方法について見ていこう．

10.3　数理計画問題

一般に，様々な問題を次の形へ書き換えることができると知られている．

入力：集合 $\boldsymbol{S} \subseteq \mathbb{R}^n$ と関数 $f : \mathbb{R}^n \to \mathbb{R}$ が与えられる．
目的：$\boldsymbol{S}$ のなかで f を最小（あるいは最大）とする元を見つける．

上で現れる関数 f を**目的関数**，集合 $\boldsymbol{S}$ のことを**実行可能集合**あるいは**実行可能領域**，その元 $\boldsymbol{x} \in \boldsymbol{S}$ を**実行可能解**という．また，集合 $\boldsymbol{S}$ を定義するときに現れる条件を，**制約条件**という．そして，実行可能解のなかで目的関数を最小（あるいは最大）とするものを**最適解**という．このような問題を総称して，**数理計画問題**という．

線形計画問題

数理計画問題の中でも制約条件がいくつかの 1 次不等式や等式であり，なおかつ目的関数が 1 次関数である問題を**線形計画問題**という．ここでは，制約条件の不等式はすべて等号付きである問題についてのみ考えるものとする．線形計画問題は，次の形へと変形できることが知られている．

(P) 線形計画問題の標準形

入力：実行可能集合 $\boldsymbol{S} = \{\boldsymbol{x} \in \mathbb{R}^n \mid A\boldsymbol{x} = \boldsymbol{b},\ \boldsymbol{x} \leq 0\}$ と目的関数 $f(\boldsymbol{x}) = \boldsymbol{c}^{\mathrm{T}}\boldsymbol{x}$ が与えられる．
目的：$\boldsymbol{S}$ のなかで f を最小とする元を見つける．

主張 10.3
線形計画問題は標準形へと変形できる．

証明
なぜこのような標準形へと変形できるのか，その理由を説明しよう．まず，最大化問題であった場合は目的関数に -1 をかけることで最小化問題にできる．次に，符号の制約がない変数 x_i があった場合，変数 x_i', x_i'' を新たに導入して $x_i = x_i' - x_i'', x_i' \geq 0, x_i'' \geq 0$ と置き換えればよい．最後に，$\sum_{i=1}^{n} a_i x_i \geq b$（または $\sum_{i=1}^{n} a_i x_i \leq b$）という形の制約があった場合は新たに変数 x_{n+1} を導入して，$\sum_{i=1}^{n} a_i x_i - x_{n+1} = b, x_{n+1} \leq 0$（または $\sum_{i=1}^{n} a_i x_i + x_{n+1} = b, x_{n+1} \leq 0$）とできる．以上の操作をすることで，標準形へと変形できる． □

また，線形計画問題が与えられたとき，その**双対問題**というものを考えることができる．先ほど与えた線形計画問題の標準形に対して，双対問題を次の形で定義する．

(D) 双対問題

入力：実行可能集合 $\boldsymbol{S}' = \{\boldsymbol{y} \in \mathbb{R}^n \mid A^{\mathrm{T}}\boldsymbol{y} \leq \boldsymbol{c}\}$ と目的関数 $g(\boldsymbol{y}) = \boldsymbol{b}^{\mathrm{T}}\boldsymbol{y}$ が与えられる．
目的：$\boldsymbol{S}'$ のなかで g を最大とする元を見つける．

双対問題の双対問題は，主問題となる．しばしば主問題は (P)，双対問題は (D) で表される．双対問題に関する有用な定理を示そう．

定理 10.1 弱双対定理
(P) と (D) それぞれの任意の実行可能解 $\boldsymbol{x}, \boldsymbol{y}$ に対して，常に不等式 $\boldsymbol{c}^{\mathrm{T}}\boldsymbol{x} \geq \boldsymbol{b}^{\mathrm{T}}\boldsymbol{y}$ が成り立つ．

証明

主問題，双対問題のそれぞれの制約条件より $\boldsymbol{c}^{\mathrm{T}}\boldsymbol{x} \geq (A^{\mathrm{T}}\boldsymbol{y})^{\mathrm{T}}\boldsymbol{x} = \boldsymbol{y}^{\mathrm{T}}\boldsymbol{b}$ □

系 10.1

(P) の任意の実行可能解 $\boldsymbol{x}$ に対して次の不等式が成立する．

$$\boldsymbol{c}^{\mathrm{T}}\boldsymbol{x} \geq (D) \text{ の最大値}$$

系 10.2

(D) の任意の実行可能解 $\boldsymbol{y}$ に対して次の不等式が成立する．

$$\boldsymbol{b}^{\mathrm{T}}\boldsymbol{y} \leq (P) \text{ の最小値}$$

系 10.3

(P) の実行可能解 $\boldsymbol{x}$ と (D) の実行可能解 $\boldsymbol{y}$ が，$\boldsymbol{c}^{\mathrm{T}}\boldsymbol{x} = \boldsymbol{b}^{\mathrm{T}}\boldsymbol{y}$ を満たすとき，$\boldsymbol{x}, \boldsymbol{y}$ はそれぞれ最適解．

定理 10.2 双対定理

(P) または (D) の一方が最適解をもてば他方も最適解をもち，(P) の最小値と (D) の最大値は等しい．

双対定理を示す前に，線形計画問題の最適解を得るアルゴリズムについて書く．