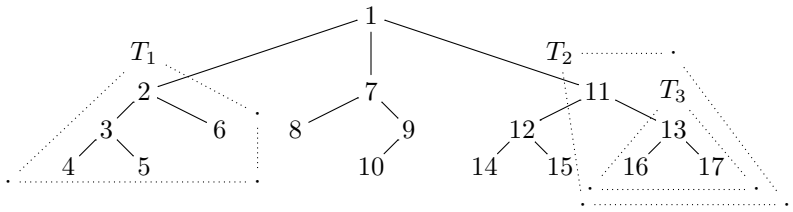
# 12　グラフ理論 (2) – 順序木と最小全域木 –

**• 12-1 ： 根付き木の順序構造**

$G = (V, E, \psi)$ をグラフとする．$G$ からいくつかの頂点や辺を取り除いて得られるものが再びグラフをなすとき，これを $G$ の **部分グラフ** という．正確に記述すると，$G' = (V', E', \psi')$ が $G$ の部分グラフであるとは，$V' \subset V$, $E' \subset E$ であって，接続関数 $\psi' : E' \to \{\{u, v\} \mid u, v \in V'\}$ が，任意の $e \in E'$ に対して $\psi'(e) = \psi(e)$ を満たすときを言う．木 $T$ の連結な部分グラフはやはり木となり，これを $T$ の **部分木** と呼ぶ．木は部分木から構成されており，部分木は木であったから，木は帰納的な構造を持っている．

$(T, 1) =$




$u$ と $v$ が辺で繋がっていて，$\mathsf{depth}_r(u) = \mathsf{depth}_r(v) - 1$ であるとき，$u$ を $v$ の **親** といい，$v$ を $u$ の **子** と呼ぶ．このとき，$T$ の頂点集合 $V$ 上の **親子関係** を

$$u > v \quad \overset{\text{def}}{\iff} \quad u \text{ は } v \text{ の親である．}$$

と定義しよう．この $>$ の反射的かつ推移的閉包を $\geq^*$ と表すことにする．$T$ の頂点 $u, v$ に対して，$u \geq^* v$ であるとき，$u$ を $v$ の **祖先**，$v$ を $u$ の **子孫** と呼ぶ．要は，$u$ が $v$ 自身であるか，$v$ よりも上位の頂点のときに，$u$ を $v$ の先祖といい，その逆の関係が子孫である．この祖先・子孫関係 $\geq^*$ は $T$ の頂点集合上の半順序を定める[*1]．

**例 12-1**　上記で与えた根付き木 $(T, 1)$ において，すべての頂点は 1 の子孫である．1 の子は頂点 $2, 7, 11$ であり，7 の子は $8, 9$ である．2 の子孫は $2, 3, 4, 5, 6$ であり，頂点 9 と頂点 13 には祖先・子孫の関係はない．

**• 12-2 ： 順序木**

根付き木 $(T, r)$ の異なる頂点 $u, v$ に対して，$u$ と $v$ が共通の親をもつとき，$u$ と $v$ は **兄弟** であるという[*2]．兄弟は $\geq^*$ では比較できない．そこで，兄弟同士に大小関係をひとつ固定して，上位の兄弟頂点の子孫は，下位の兄弟頂点の子孫よりも上位になるように定める．つまり，$T$ の頂点全体に全順序 $\geq^{\text{tot}}$ を次のように定める：

$$u \geq^{\text{tot}} v$$
$$\overset{\text{def}}{\iff} \begin{cases} u \geq^* v & (u \text{ と } v \text{ は } \geq^* \text{ で比較可能．}) \\ u \text{ の祖先 } u' \text{ と } v \text{ の祖先 } v' \text{ が兄弟で，} u' \text{ が } v' \text{ より上位である．} & (u \text{ と } v \text{ は } \geq^* \text{ で比較不可能．}) \end{cases}$$

このようにして，兄弟同士にある順序を用いて $\geq^{\text{tot}}$ を導入した根付き木を **順序木** という．順序木を描くとき，左側に上位の頂点を，右側に下位の頂点を配置する．

**例 12-2**　上記で与えた根付き木 $(T, 1)$ を順序木とみなしたとき，頂点 $i, j$ に対して

$$i \geq^{\text{tot}} j \quad \iff \quad i \leq j$$

である．ここで，右辺は自然数の間の自然な順序である．例えば，頂点 5 と頂点 17 を比較すると，5 の祖先 2 と，17 の祖先 11 は兄弟であり，2 が 11 の左側にあるので $2 \geq^{\text{tot}} 11$ である．よって，$5 \geq^{\text{tot}} 17$ である．

[*1] **演習問題 12-1**
[*2] $v$ 自身を $v$ の兄弟とは呼ばない．

## • 12-3 : 最小全域木

**定義 12.1.** 有限グラフ $(V, E, \psi)$ および **重み関数** と呼ばれる写像 $w : E \longrightarrow \mathbb{R}$ からなるような 4 つ組 $G = (V, E, \psi, w)$ を **重み付きグラフ** と呼ぶ. 辺 $e \in E$ に割り当てられている実数 $w(e) \in \mathbb{R}$ を $e$ の **重み** と呼ぶ. 重み付きグラフ $G$ の **重み** とは, 全ての辺に割り当てられている重みの総和

$$w(G) := \sum_{e \in E} w(e)$$

で定義する. ただし, 右辺は全ての辺 $e \in E$ についての $w(e)$ の総和である.

つまり, グラフの各辺 $e \in E$ にそれぞれ実数 $w(e) \in \mathbb{R}$ が割り当てられているようなグラフである. 辺の重みをすべて形式的に 1 と定めれば, 辺の間に重みに関する優劣はなくなるので通常の意味での有限グラフと同一視できることに注意しておこう.

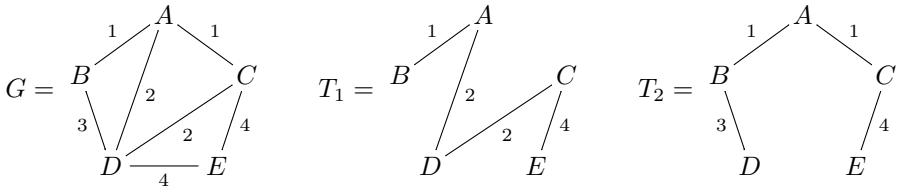
重み付きグラフ $G = (V, E, \psi, w)$ に対して, $G$ からいくつかの辺を取り除いて得られる重み付きグラフを $G$ の **全域部分グラフ** という. 正確には次の通りである. $E'$ を $E$ の部分集合として, これに対して新たな接続関数と重み関数を

$$\psi' : E' \longrightarrow \{\{x, y\} \mid x, y \in V\}; \quad e \longmapsto \psi(e)$$
$$w' : E' \longrightarrow \mathbb{R}; \quad e \longmapsto w(e)$$

で定義する. こうして重み付きグラフ $(V, E', \psi', w')$ を構成できたが, これが全域部分グラフである[*3].

**定義 12.2.** 有限重み付きグラフ $G = (V, E, \psi, w)$ の全域部分グラフ $T' = (V, E', \psi', w')$ について, $(V, E', \psi')$ が木であるとき, $T'$ を **全域部分木** と呼ぶ. 全域部分木の中で, 重みが最小となるものを **最小全域木** という[*4].

**例 12-3**　重み付きグラフ $G$ が以下のように与えられているとする. ただし, 辺に対応する数は, その辺の重みである. このとき, $G$ の重みは 17 である. また, $T_1$, $T_2$ は $G$ の全域部分木で, その重みは共に 9 である.




一般に最小全域木を求めることはとても難しい. 自明な方法としては, 全ての全域部分木を求めて, これらの重みを計算して最も重みが小さいものを選べば原理的には求めることができる. しかし, このような全探索の方法は非効率であるし, グラフによっては途方もない計算量を要するので, 大変効率が悪い. これに対して, 最小全域木は次に述べる貪欲アルゴリズム（greedy algorithm）によって求めることができる.

**アルゴリズム 12.3** (Kruskal のアルゴリズム)**.** 最小全域木を求めるアルゴリズムを記述する.
(**Input**)　連結な重み付きグラフ $G = (V, E, \psi, w)$ で $p = |V|$, $q = |E|$ とする.
(**Output**) 最小全域木 $T$

`step.0`： $E(T) = \varnothing$, $i = 1$ とする. $G$ の辺の重みを小さい順に

$$w(e_1) \leq w(e_2) \leq \cdots \leq w(e_q)$$

と並べ替える. $T$ を $E(T)$ から定まる $G$ の全域部分グラフとする.

[*3] 全域部分グラフは $E' \subset E$ を指定すればただ一つに決まる.
[*4] 最小全域木は一般には一意的には定まらない.

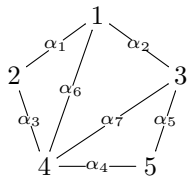
step.1： $|E(T)| \neq p-1$ ならば step.2 へ. $|E(T)| = p-1$ ならば $T$ を出力して終了する.

step.2： $e_i$ に関して, $T$ に $e_i$ を加えてもサイクルができないならば, $e = e_i$ として step.3 へ. そうでないならば, $i$ を $i+1$ に置き換えて step.2 へ.

step.3： $E(T) \cup \{e_i\}$ を新たに $E(T)$ と置き, $T$ に辺 $e$ を加えたグラフを新たに $T$ とし, $i$ を $i+1$ に置き換えて step.1 に戻る.

**例 12-4**　次の重み付きグラフ $G$ の最小全域木を Kruskal のアルゴリズムを用いて求めてみよう.

$G =$




| 辺 | $\alpha_1$ | $\alpha_2$ | $\alpha_3$ | $\alpha_4$ | $\alpha_5$ | $\alpha_6$ | $\alpha_7$ |
|---|---|---|---|---|---|---|---|
| 重み | 1 | 1 | 3 | 4 | 4 | 2 | 2 |

step.0： $E(T) = \varnothing$, $i = 1$ とする. $G$ の辺の重みを小さい順に

$$e_1 = \alpha_1, \quad e_2 = \alpha_2, \quad e_3 = \alpha_6, \quad e_4 = \alpha_7, \quad e_5 = \alpha_3, \quad e_6 = \alpha_4, \quad e_7 = \alpha_5$$

とおく.

step.1： $|E(T)| = 0 \neq 5-1 = 4$ なので継続する.

step.2： $e_1$ に関して, $T$ に $e_1$ を加えてもサイクルができないので, $e = e_1$ として step.3 へ.

step.3： $E(T) := \{e_1\}$ と改めて置き, $i = 2$ として step.1 に戻る.

step.1： $|E(T)| = 1 \neq 4$ なので継続する.

step.2： $e_2$ に関して, $T$ に $e_2$ を加えてもサイクルができないので, $e = e_2$ として step.3 へ.

step.3： $E(T) := \{e_1, e_2\}$ と改めて置き, $i = 3$ として step.1 に戻る.

step.1： $|E(T)| = 2 \neq 4$ なので継続する.

step.2： $e_3$ に関して, $T$ に $e_3$ を加えてもサイクルができないので, $e = e_3$ として step.3 へ.

step.3： $E(T) := \{e_1, e_2, e_3\}$ と改めて置き, $i = 4$ として step.1 に戻る.

step.1： $|E(T)| = 3 \neq 4$ なので継続する.

step.2： $e_4$ に関して, $T$ に $e_4$ を加えるとサイクルができるので, $i = 5$ として step.2 へ.

step.2： $e_5$ に関して, $T$ に $e_5$ を加えるとサイクルができるので, $i = 6$ として step.2 へ.

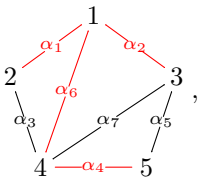
step.2： $e_6$ に関して, $T$ に $e_6$ を加えてもサイクルができないので, $e = e_6$ として step.3 へ.

step.3： $E(T) := \{e_1, e_2, e_3, e_6\}$ と改めて置き, $i = 7$ として step.1 に戻る.

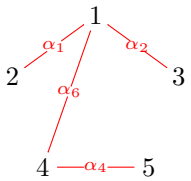
step.1： $|E(T)| = 4$ なので $T$ を出力して終了する.

以上の手続きによって, $G$ の最小全域木 $T$（重みは 8）が得られる.

$G =$




, $T =$



**小テスト 12-1**　木の構造を, 親子関係によって帰納的に定義することにより python で実装することを考える. 例えば, 以下のコードでは, 頂点 1 を根とするある根付き木を表している.

Listing 1　「木構造の例」

```
tree = {
    1: {
        2: {
            3: {},
            4: {}
        },
        5: {}
    }
}
```

次のように帰納的に定義される木について, 以下の問いに答えよ.

Listing 2　「課題の根付き木」

```

tree = {
    1: {
        2: {
            3: {
                4: {
                    5: {}, 6: {}, 7: {}
                },
                8: {},
                9: {
                    10: {}
                }
            },
            11: {}
        },
        12: {
            13: {
                15: {
                    17: {},
                    18: {}
                },
                16: {
                    19: {},
                    20: {}
                }
            },
            14: {
                21: {
                    22: {}
                }
            }
        },
        23: {
```

```
            24: {
                25: {},
                26: {},
                27: {}
            }
        }
    }
}

def print_tree(node, tree_dict, depth=0):
    print(" " * depth + str(node))
    for child in tree_dict.get(node, {}):
        print_tree(child, tree_dict[node], depth + 1)
```

(1) 8 の親はなにか.

(2) 2 と兄弟であるものをすべて選べ

(3) 各兄弟について, 通常の数の大小によって頂点の上位と下位を定義し, これにより頂点集合に全順序 $\geq^{\mathrm{tot}}$ を導入する. このとき, 12 よりも上位の頂点をすべて選べ.

(4) この木の根を頂点 26 としたとき, 深さが 5 にあるものを全て求めよ.

**小テスト 12-2**　2 つの集合 $V = \{n \in \mathbb{N} \mid 1 \leq n \leq 6\}$ と $E = \{\alpha_1, \alpha_2, \ldots, \alpha_8\}$ と以下で定まる接続関数 $\psi$ で定まるグラフ $G = (V, E, \psi)$ を考えよう.

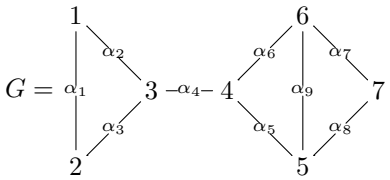
$$\psi(\alpha_1) = \{1,2\},\ \ \psi(\alpha_2) = \{1,3\},\ \ \psi(\alpha_3) = \{2,3\},\ \ \psi(\alpha_4) = \{2,4\},$$
$$\psi(\alpha_5) = \{3,4\},\ \ \psi(\alpha_6) = \{4,5\},\ \ \psi(\alpha_7) = \{5,6\},\ \ \psi(\alpha_8) = \{4,6\}.$$

グラフ $G$ の各辺の重みを以下で定義する. このとき, この重み付きグラフの最小全域木の重みを答えよ.

| 辺 | $\alpha_1$ | $\alpha_2$ | $\alpha_3$ | $\alpha_4$ | $\alpha_5$ | $\alpha_6$ | $\alpha_7$ | $\alpha_8$ |
|---|---|---|---|---|---|---|---|---|
| 重み | 3 | 1 | 7 | 5 | 2 | 7 | 3 | 8 |

**演習問題 12-1**　根付き木 $(T, r)$ において, $T$ の頂点集合 $V$ 上の祖先・子孫関係 $\geq^*$ が $V$ 上の半順序を定めることを示せ.

**演習問題 12-2**　次のグラフ $G$ に Kruskal のアルゴリズムを適用することで最小全域木を求めよ.




| 辺 | $\alpha_1$ | $\alpha_2$ | $\alpha_3$ | $\alpha_4$ | $\alpha_5$ | $\alpha_6$ | $\alpha_7$ | $\alpha_8$ | $\alpha_9$ |
|---|---|---|---|---|---|---|---|---|---|
| 重み | 1 | 2 | 5 | 8 | 2 | 3 | 1 | 5 | 4 |

**演習問題 12-3**　次の重み付きグラフに Kruskal のアルゴリズムを適用して, 最小全域木となるものをすべて求めよ. ただし, 各辺に対応する数はその重みである.