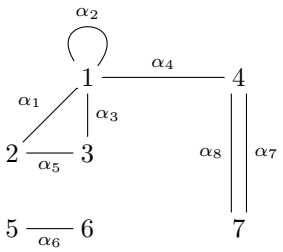
# 12　離散グラフ理論 （1） –木と最小全域木–

**• 12-1 : 無向グラフ**

第 9 章で「有向グラフ」については扱ったが, 各辺における向きを無視したものが「(無向) グラフ」である. 多くの現実的な状況では, 点の集合とそれらの点の対のいくつかを線分で結んだ図形によって簡単に表現できる. 例えば, 頂点が人間を表し, それらの交友関係を線で結ぶことができる. あるいは, 点がインターネットサーバーで, それらのサーバー同士が回線で結ばれている状況を線分で結ぶことでインターネットの模式図を表すこともできる. これらの数学的抽象化がグラフの概念となっている. 以下の図はグラフの一例である.




では, グラフの正確な定義を与えよう.

**定義 12.1.** 2 つの集合 $V, E$ が $V \cap E = \varnothing$ であるとする. 集合 $V, E$ と **接続関数** とよばれる写像 $\psi : E \to \{\{x, y\} \mid x, y \in V\}$ の 3 つ組 $G = (V, E, \psi)$ を **(無向) グラフ** という. $V$ の要素をグラフ $G$ の **頂点**, $E$ の要素をグラフ $G$ の **辺** と呼ぶ. 辺 $e \in E$ に対して, $\psi(e) = \{x, y\}$ であるとき, $e$ は $x$ と $y$ に **接続している** という. $V$ と $E$ が有限集合であるとき, $G$ を **有限グラフ** と呼ぶ.

グラフ $G = (V, E, \psi)$ の接続関数 $\psi$ は, 辺の端点を指定する写像である. つまり, 辺 $e \in E$ に対して, $\psi(e) = \{x, y\}$ となるような頂点 $x, y \in V$ (ただし, $x = y$ の可能性もある. ) が定まるが, このときこれを

$$x \overset{e}{\text{———}} y \qquad \text{もしくは} \qquad x \text{——}e\text{——} y$$

という風に図示する. $\psi(e) = \{x\}$ であるとき, $e$ を $x$ を頂点に持つ **ループ** という.

**例 12-1**　冒頭のグラフは, $V = \{1, 2, \ldots, 7\}$, $E = \{\alpha_1, \alpha_2, \ldots, \alpha_8\}$ であって, 接続関数は

$$\psi(\alpha_1) = \{1, 2\}, \quad \psi(\alpha_2) = \{1\}, \quad \psi(\alpha_3) = \{1, 3\}, \quad \psi(\alpha_4) = \{1, 4\},$$
$$\psi(\alpha_5) = \{2, 3\}, \quad \psi(\alpha_6) = \{5, 6\}, \quad \psi(\alpha_7) = \{4, 7\}, \quad \psi(\alpha_8) = \{4, 7\}$$

で与えられる.

グラフの頂点を「地点」, 辺を「2 つの地点を結ぶ道路」だと思えば, 図示されたグラフに対して「ある地点から辺をたどってどこかの地点へ移動する道」を考えることができる. 例えば, 冒頭のグラフにおいて, 頂点 1 から頂点 7 へ移動する辺のたどり方を考えてみると

- $\alpha_1 \to \alpha_5 \to \alpha_3 \to \alpha_2 \to \alpha_4 \to \alpha_8$
- $\alpha_2 \to \alpha_4 \to \alpha_7$
- $\alpha_2 \to \alpha_3 \to \alpha_5 \to \alpha_1 \to \alpha_2 \to \alpha_2 \to \alpha_4 \to \alpha_7$

など様々ある. 従って, 頂点 1 と頂点 7 は「繋がっている」と自然に理解できるだろう.


担当：茨城大学 情報科学領域　宮本 賢伍

一方, 頂点 1 から頂点 5 への移動はどうだろうか. このときは, 辺をたどって頂点 1 から頂点 5 に移動できないので, 「繋がっていない」と理解できる. このように, 2 つの地点が繋がっているか繋がっていないかは, 辺をたどって移動できるかできないかで理解できる. 図示できるような簡単なグラフならば, 見るだけで繋がっているかを簡単に把握できるが, ネットワークのような大規模なグラフだと一見するとわからない場合がある. そこで, この「辺をたどる」という操作をきちんと数学的に定義しよう.

**定義 12.2.** グラフ $G = (V, E, \psi)$ に対して, 頂点 $v_0, v_1, \ldots, v_n$ と辺 $e_1, e_2, \ldots, e_n$ の列

$$(v_0, e_1, v_1, e_2, v_2, \ldots, v_{n-1}, e_n, v_n)$$

であって, 任意の $i = 1, \ldots, n$ に対して $\psi(e_i) = \{v_{i-1}, v_i\}$ が成り立つようなものを, $v_0$ と $v_n$ を結ぶ **歩道** といい, $n$ をこの歩道の **長さ** と呼ぶ. $v_0 = v_n$ となるような歩道は **閉じている** という. 同じ頂点を含まない歩道を **道**, 同じ辺を含まない歩道を **小道** という. 閉じている道を **閉路**, 閉じている小道を **回路** と呼ぶ. グラフ $G$ の異なる 2 つの頂点の間を結ぶ歩道が存在するときに, $G$ は **連結である** という.

**レポート 12-1**　**例 12-1** で与えられたグラフ $G$ において, 頂点 1 からはじまる長さが 4 であるような閉じている道を全て求めよ. また, 頂点 1 からはじまる長さが 4 であるような閉路を全て求めよ.

### • 12-2 : 木

**定義 12.3.** 閉路を含まないような連結なグラフを **木** という.

証明は行わないが, グラフが木であることの必要十分条件はたくさんある. 感覚的に理解できるものもあるだろう.

**命題 12.4.** 頂点の数が $n$ であるようなグラフ $T = (V, E, \psi)$ に対して, 以下の命題は同値である.

(a) $T$ は木である.
(b) $T$ の異なる 2 つの頂点 $x, y \in V$ に対して, $x$ と $y$ を結ぶ道がただ一つ存在する.
(c) $T$ は連結であり, $n - 1$ 個の辺からなる.
(d) $T$ は閉路のないグラフであり, $n - 1$ 個の辺からなる.
(e) 任意の $e \in E$ を $T$ から取り除いても, グラフ $T' = (V, E \setminus \{e\}, \psi')$ は非連結である. ただし, $\psi'$ は任意の $f \in E \setminus \{e\}$ に対して $\psi'(f) = \psi(f)$ であるような接続関数である.
(f) $T$ に, どの辺 $e \in E$ でも接続されていないような任意の 2 つの頂点 $x, y \in V$ に対して, $T$ に $x, y$ を接続する新たな辺を加えたものにはただ一つの閉路が存在する.

**レポート 12-2**　頂点の数が 6 個であるような木を全て求めよ. （そのような木は 6 個存在する. ）

### • 12-3 : 重み付きグラフ

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. 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 $e \in E$ 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. 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.


担当：茨城大学 情報科学領域　宮本 賢伍

- **12-4 : 最小全域木**

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. 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. 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. 最小全域木は一般には一意的には定まらない.

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. しかし, 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ことが知られている.

**アルゴリズム 12.5** (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

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

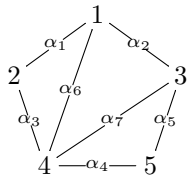
と並べ替える. $T$ を $E(T)$ から定まる $G$ の全域部分グラフとする.

`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 $e_i$ を加えても閉路ができないならば, $e=e_i$ 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 12-2**　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.

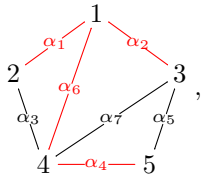
`step.1`： $|E(T)|=0 \neq 5-1=4$ なので継続する.

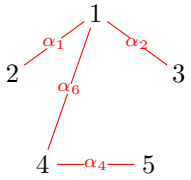
`step.2`： $e_1$ に関して, $T$ に $e_1$ を加えても閉路ができないので, $e=e_1$ として `step.3` へ.

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

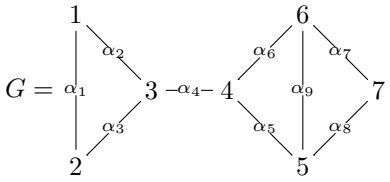
step.1： $|E(T)| = 1 \neq 4$ なので継続する.
step.2： $e_2$ に関して, $T$ に $e_2$ を加えても閉路ができないので, $e = e_2$ として step.3 へ.
step.3： $E(T) := \{e_1, e_2\}$ と改めて置き, $i = 3$ として step.1 に戻る.
step.1： $|E(T)| = 2 \neq 4$ なので継続する.
step.2： $e_3$ に関して, $T$ に $e_3$ を加えても閉路ができないので, $e = e_3$ として step.3 へ.
step.3： $E(T) := \{e_1, e_2, e_3\}$ 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 $e_4$ を加えると閉路ができるので, $i = 5$ として step.2 へ.
step.2： $e_5$ に関して, $T$ に $e_5$ を加えると閉路ができるので, $i = 6$ として step.2 へ.
step.2： $e_6$ に関して, $T$ に $e_6$ を加えても閉路ができないので, $e = e_6$ 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 $T$ が得られる.

$G =$ 

, $T =$ 


**レポート 12-3**　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.

$G =$ 


| 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 |


担当：茨城大学 情報科学領域　宮本 賢伍