11 グラフ理論 (1) – グラフと木 –

- **11-1 : 無向グラフ**

現実世界の多くの状況では, 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．

定義 11.1. 2 つの集合 V, E が $V \cap E = \varnothing$ であるとする．E の各元に対して, V の元の非順序対を対応させる **接続関数** とよばれる写像

$$\psi : E \to \{\{x, y\} \mid x, y \in V\}$$

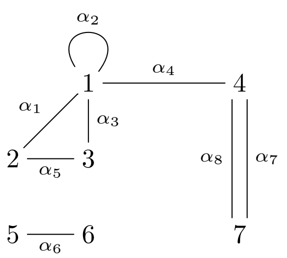
の 3 つ組 $G = (V, E, \psi)$ を **（無向）グラフ** という．V の要素をグラフ G の **頂点**, E の要素をグラフ G の **辺** と呼ぶ．辺 $e \in E$ に対して, $\psi(e) = \{x, y\}$ であるとき, e は x と y に **接続している** という．V と E が有限集合であるとき, G を **有限グラフ** と呼ぶ．

グラフ $G = (V, E, \psi)$ の接続関数 ψ は, 辺がどの頂点を結んでいるか, 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．このとき, 頂点 x, y を辺 e の **端点** と呼ぶ．$\psi(e) = \{x\}$ であるとき, e を x を頂点に持つ **ループ** という．

例 11-1 次のグラフ



は, $V = \{1, 2, \ldots, 7\}$, $E = \{\alpha_1, \alpha_2, \ldots, \alpha_8\}$ であって, 接続関数は

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

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, **例 11-1** 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．このとき, 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．

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 11.2. 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

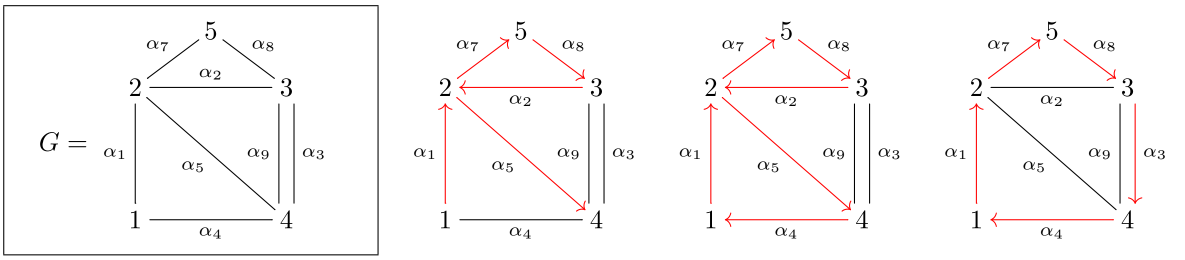
$$w = (v_0, e_1, v_1, e_2, v_2, \ldots, v_{n-1}, e_n, v_n)$$

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．$n = 0$ のとき, **自明な道** といい, これは頂点と同一視される．道 w に対して, v_0 を w の **始点**, v_n を w の **終点** という．グラフ G の任意の 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．

歩道の中でも, 特別なものには名前がついている．

用語	定義	意味
w が **小道**	同じ辺を高々 1 回しか通らない歩道	同じ辺を 2 回以上通らずに辿ったもの
w が **道**	始点と終点のみが一致する場合を除いて, 同じ頂点を高々 1 回しか通らない歩道	同じ頂点を 2 回以上通らずに辿ったもの
w が **回路**	閉じている小道	同じ辺を 2 回以上通らずに, 始点に帰ってくるように辿ったもの
w が **閉路**	閉じている道	同じ頂点を 2 回以上通らずに, 始点に帰ってくるように辿ったもの

例 11-2 次のグラフ G について考えよう．



(1) $(1, \alpha_1, 2, \alpha_7, 5, \alpha_8, 3, \alpha_2, 2, \alpha_5, 4)$ は同じ辺を通っていないが, 頂点 2 を 2 度通っているから道ではない．
(2) $(1, \alpha_1, 2, \alpha_7, 5, \alpha_8, 3, \alpha_2, 2, \alpha_5, 4, \alpha_4, 1)$ は回路だが, 頂点 2 を 2 度通っているから閉路ではない．
(3) $(1, \alpha_1, 2, \alpha_7, 5, \alpha_8, 3, \alpha_3, 4, \alpha_4, 1)$ は回路であり, 2 度通っている頂点 1 は始点かつ終点なので閉路でもある．

歩道は, **他に解釈の仕様がない限り,** 頂点だけの列や辺だけの列で表示しても良い．例えば,

$$(1, \alpha_1, 2, \alpha_7, 5, \alpha_8, 3, \alpha_2, 2, \alpha_5, 4) = (1, 2, 5, 3, 2, 4) = (\alpha_1, \alpha_7, \alpha_8, \alpha_2, \alpha_5)$$

と表示できる．一方で, $(1, \alpha_1, 2, \alpha_7, 5, \alpha_8, 3, \alpha_3, 4, \alpha_4, 1)$ は, $(\alpha_1, \alpha_7, \alpha_8, \alpha_3, \alpha_4)$ と表示してもよいが,

$$(1, 2, 5, 3, 4, 1)$$

だと 3 から 4 への辿り方が曖昧になるので, この表示は不合理である．

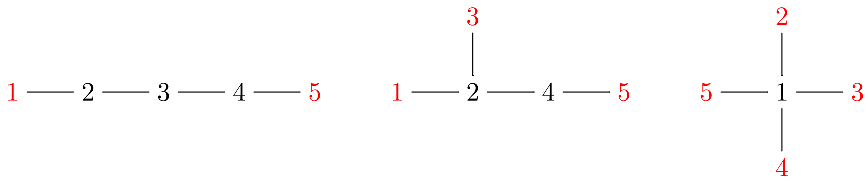
• 11-2：木

定義 11.3. (1) 回路でも閉路でもないような閉じた歩道を **サイクル** と呼ぶ.

(2) サイクルを含まないような連結なグラフを **木** という[*1].

(3) 木の頂点 v に接続している辺が高々 1 本であるとき, v を **葉** という.

例 11-3 頂点の数が 5 であるような木は, 次の形のものしかない. また, 赤字で示した頂点が葉である.



命題 11.4. $T = (V, G, \psi)$ を木とする. このとき, 次が成り立つ.

(1) 任意の相異なる 2 頂点 $u, v \in V$ に対して, u から v への道がただ一つ存在する.

(2) $|V| \geq 2$ ならば, T には少なくとも 2 つの葉が存在する.

(3) $|V| = |E| + 1$ が成り立つ.

証明. (1) T は連結で閉路がないので, 相異なる 2 頂点の間の歩道はただ一つしかない.

(2) $|V| = 2$ ならば, $T =$ 1 —— 2 という形をしている. このとき, 頂点 $1, 2$ は葉である.

次に $|V| > 2$ の場合は背理法で証明しよう. 葉が 1 つの場合は, 木は連結なので $|V| = 1, E = \varnothing$ であり, これは仮定に反する. 従って, 葉は少なくとも 2 つ存在してなければならない.

(3) $v \in V$ を T の葉とすれば, v に接続している辺は高々 1 本である. v に辺が接続されていないとき, T は頂点が v のみで辺を持たない木である. このときは $|V| = 1$, $|E| = 0$ なので主張は正しい. 次に, v に接続するただ一つの辺 e が存在するとき, 新たな木 $T' := (V', E', \psi')$ を, $V' := V \setminus \{v\}$, $E' := E \setminus \{e\}$, $\psi' = \psi|_{E'}$ とおくと, これはまた木である. この木 T' の葉 v' について, これまでの議論と同様にして, v' とそれに接続されている唯一の辺 e' を取り除いて新たな木を構成する. この操作を, ただ一つの頂点のみからなる木になるまで繰り返したとき, これまで消去してきた頂点の数と辺の数は等しいので $|V| = |E| + 1$ が成り立つ. □

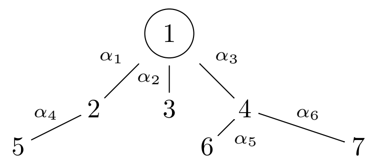
• 11-3: 根付き木

定義 11.5. 木 T に対して, 頂点 r を一つ指定したとき, 組 (T, r) を **根付き木** という. T の頂点 v に対して, r から v への道の長さを, v の **深さ** と呼び, これを $\mathsf{depth}_r(v)$ と表す.

根付き木 (T, v) は, 根を最上位に, そこから伸びる辺を下方に向かわせ, 葉を最下位になるように配置して描くのが一般的である.

例 11-4 図のような根付き木 $(T, 1)$ において, $2, 3, 4$ は深さ 1 の頂点, $5, 6, 7$ は深さ 2 の頂点である.

$(T, 1) =$

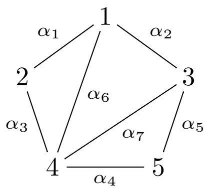


すなわち, $\mathsf{depth}_1(2) = \mathsf{depth}_1(23 = \mathsf{depth}_1(4) = 1$ であり, $\mathsf{depth}_1(5) = \mathsf{depth}_1(6) = \mathsf{depth}_1(7) = 2$ である.

[*1] 教科書では, “閉路含まない連結なグラフ”を木と定義しているが, これは誤りである.

小テスト 11-1 次のグラフ G について考える.

$G =$



(1) 選択肢のなかから小道であるものを全て選べ.

(2) 選択肢のなかから道であるものを全て選べ.

(3) 選択肢の中から閉路であるものを全て選べ.

小テスト 11-2 2 つの集合 $V = \{n \in \mathbb{N} \mid 1 \leq n \leq 8\}$ と $E = \{\alpha_1, \alpha_2, \ldots, \alpha_7\}$ と以下で定まる接続関数 ψ で定まる木 $T = (V, E, \psi)$ について, 以下の問いに答えよ.

$$\psi(\alpha_1) = \{4, 8\},\quad \psi(\alpha_2) = \{4, 7\},\quad \psi(\alpha_3) = \{3, 4\},\quad \psi(\alpha_4) = \{3, 5\},$$
$$\psi(\alpha_5) = \{3, 6\},\quad \psi(\alpha_6) = \{1, 3\},\quad \psi(\alpha_7) = \{1, 2\}$$

(1) T の葉を全て答えよ.

(2) 根付き木 $(T, 1)$ において, 最も深さが大きい頂点を全て答えよ.

(3) 根付き木 $(T, 8)$ において, 最も深さが大きい頂点を全て答えよ.

(4) 根付き木 $(T, 4)$ において, 最も深さが大きい頂点を全て答えよ.

演習問題 11-1 頂点の数が 6 個であるような木を全て求めよ.

演習問題 11-2 各頂点に接続する辺の数が 1 または 3 であるような木の頂点数が n であるとする. このとき, 3 本の辺に接続する頂点の数を n を用いて表せ. （ヒント：握手定理を用いる. ）

演習問題 11-3 木 T において, U を 3 本以上の辺に接続している頂点全体の集合とする. $x \in U$ に対して, x に接続している辺の数を $d(x)$ で表すとき, T の葉の数は

$$\sum_{x \in U} (d(x) - 2) + 2$$

であることを示せ.

- **11-A: 2 分木リスト構造**

情報工学科なので, 木がどういう場合に有効であるかを紹介する．リストはコンピュータでは可変な大きさのデータを扱うときに用いられるデータ構造である．コンピュータ・プログラミングにおける基本的なデータ構造として **リスト構造** がある．この, リスト構造を表すときには, デリミタとしてカンマではなく, 縦棒を用いている．リスト構造自体は次のように再帰的に定義される．

(基礎) nil はリスト（空リスト）である．
(帰納) x がアトムで, A, B がリストであれば, $(x|A)$, $(A|x)$, $(A|B)$ はリストである．
(限定句) 以上の手続きを有限回適用して構成されるものだけがリストである．

この定義ではリストは 2 分木構造（つまり, 葉を除く全ての頂点に接続している辺の数がちょうど 2 本）となっている．デリミタとしてカンマを使うリストは分岐の数を制限しない表現である．また, カンマを使った任意のリストは, 縦棒を使う 2 分木のリストで表すことができる．

いくつか具体例をみてみよう．

(1) $1, 2, 3$ をアトムとする．$(1|(2|(3|\mathrm{nil})))$ はリストである．これの意味は次の通り．
  (a) nil はリストである．
  (b) アトム 3 に対して, $(3|\mathrm{nil})$ はリストである．これは, 空リスト nil に 3 を追加することを表す．
  (c) アトム 2 に対して, $(2|(3|\mathrm{nil}))$ はリストである．これは, 2 をリストの前方に追加することを表す．
  (d) アトム 1 に対して, $(1|(2|(3|\mathrm{nil})))$ はリストである．これは, 1 をリストの前方に追加することを表す．

  結果として, $(1|(2|(3|\mathrm{nil}))) = (1, 2, 3)$ となる．

(2) a, b をアトムとする．$((a|\mathrm{nil})|(b|\mathrm{nil}))$ はリストである．これの意味は次の通り．
  (a) nil はリストである．
  (b) アトム a に対して, $(a|\mathrm{nil})$ はリストである．これは, 空リスト nil に a を追加することを表す．
  (c) アトム b に対して, $(b|\mathrm{nil})$ はリストである．これは, 空リスト nil に b を追加することを表す．いま, 結果としては (a), (b) という 2 つのリストが出来上がっている．
  (d) $((a|\mathrm{nil})|(b|\mathrm{nil}))$ はリストである．これは, $(b|\mathrm{nil})$ の前方に $(a|\mathrm{nil})$ を追加することを表す．

  結果として, $((a), b)$ を得る．

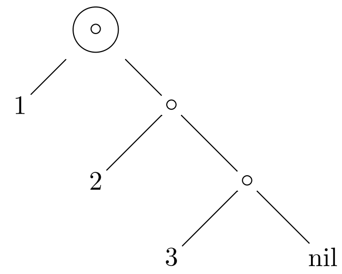
(3) a, b, c をアトムとする．$((a|(b|\mathrm{nil}))|(c|\mathrm{nil}))$ はリストである．これの意味は次の通り．
  (a) nil はリストである．
  (b) アトム c に対して, $(c|\mathrm{nil})$ はリストである．これは, 空リスト nil に c を追加することを表す．
  (c) アトム b に対して, $(b|\mathrm{nil})$ はリストである．これは, 空リスト nil に b を追加することを表す．いま, 結果としては (b), (c) という 2 つのリストが出来上がっている．
  (d) $(a|(b|\mathrm{nil}))$ はリストである．これは, $(b|\mathrm{nil})$ の前方に a を追加することを表す. いま, 結果としては (a, b), (c) という 2 つのリストが出来上がっている．
  (e) $((a|(b|\mathrm{nil}))|(c|\mathrm{nil}))$ はリストである. これは, $(b|\mathrm{nil})$ の前方に $(a|(b|\mathrm{nil}))$ を追加することを表す．

  結果として, $((a, b), c)$ を得る．

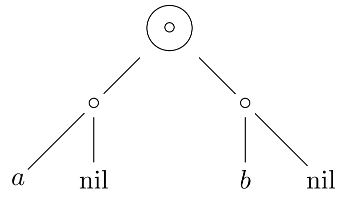
プログラミング言語 LIPS では, **セル** と呼ばれる基本構造を多用する．これは次のセルを指すポインタを 2 つもっており, セルを頂点, ポインタを辺とみなせば分岐が丁度 2 であるような木に対応する．一般的なセルから成る構造は親のセルが複数あることも許すので, 閉路を許すグラフになることに注意しよう．リスト構造で, ただ 1 つの **開始セル** をもち, 開始セル以外の全てのセルについて親のセルは唯一つに限り, かつ親以外の祖先へのポインタを禁止する（つまり上位にアクセスできないようにする）ことで, 開始セルを根とする 2 分木構

造とできる．上の 3 つの例に対応する根付き二分木は次のようになる．

$((1|(2|(3|\mathrm{nil})))$:



$((a|\mathrm{nil})|(b|\mathrm{nil}))$:



$((a|(b|\mathrm{nil}))|(c|\mathrm{nil}))$: