グラフ

離散数学・オートマトン

2020年後期

佐賀大学理工学部　只木進一

グラフとは

- 日常用語ではネットワーク
  - インターネット
  - ヒトの繋がり
  - 交通網
  - 作業手順
- 要素の繋がり方に注目

グラフの定義

- グラフ$G = (V, E)$
  - 頂点(node)の集合V
  - 辺(edge)の集合E
    - 頂点uとvを結ぶ辺$e = (u, v)$
    - 頂点uとvを辺eの端点という

グラフの定義

- 無向グラフ(non-directed graphs)
  - 辺に向きの無いグラフ
  - $\partial : E \to V \times V$: 辺から端点への写像
- 有向グラフ(directed graphs)
  - 辺に向きの有るグラフ
  - $\partial^{+} : E \to V$: 始点、$\partial^{-} : E \to V$: 終点

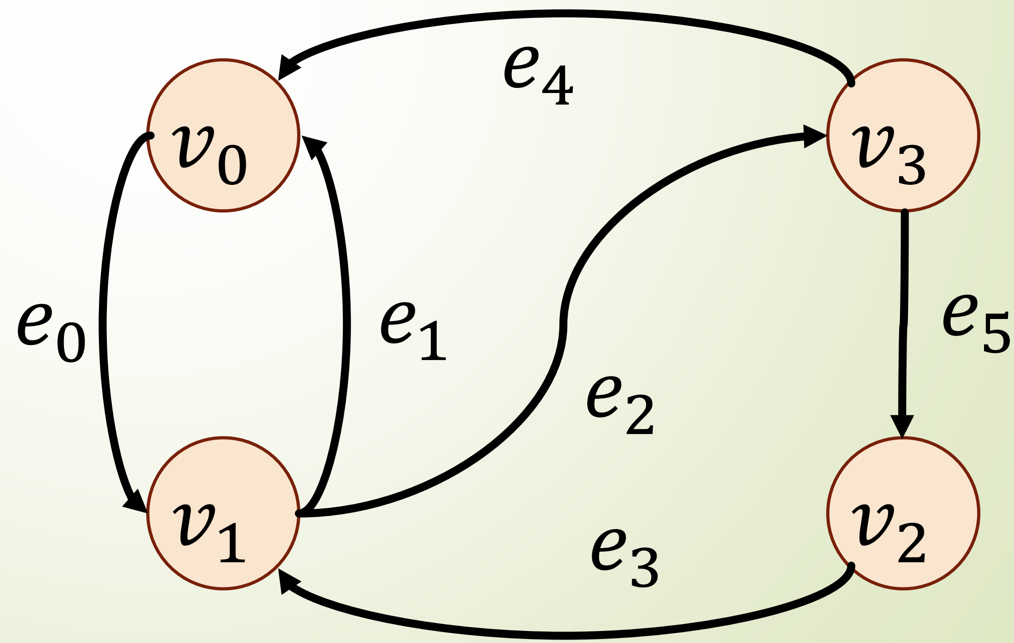
$$V = \{v_0, v_1, v_2, v_3\}$$

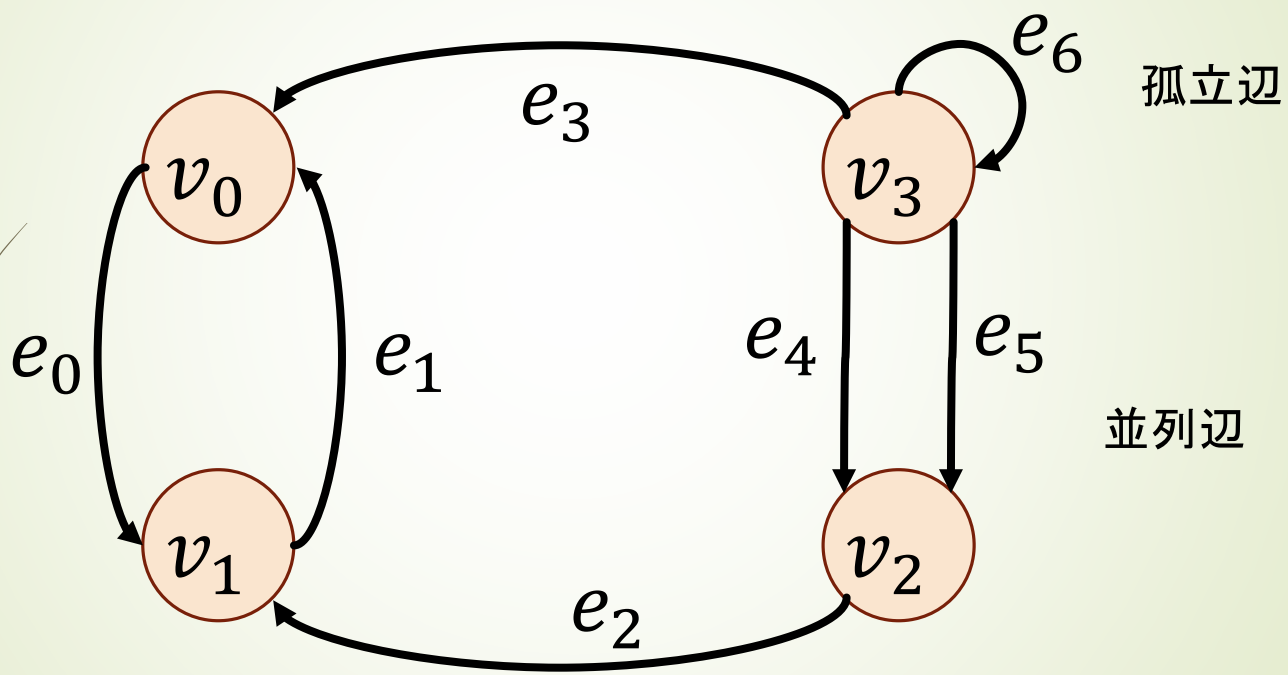
$$E = \{e_0, e_1, e_2, e_3.e_4, e_5\}$$

$$\partial^+ e_0 = v_0 \quad \partial^- e_0 = v_1 \quad \partial^+ e_1 = v_1 \quad \partial^- e_1 = v_0$$

$$\partial^+ e_2 = v_1 \quad \partial^- e_2 = v_3 \quad \partial^+ e_3 = v_2 \quad \partial^- e_3 = v_1$$

$$\partial^+ e_4 = v_3 \quad \partial^- e_4 = v_0 \quad \partial^+ e_5 = v_3 \quad \partial^- e_5 = v_2$$





e_6
孤立辺
e_3
v_0
v_3
e_0
e_1
e_4
e_5
並列辺
v_1
v_2
e_2

グラフの定義2

- 無向グラフ(non-directed graphs)
  - $\delta : V \to 2^E$: 頂点から辺の集合
- 有向グラフ(directed graphs)
  - $\delta^+ : V \to 2^E$: 頂点を始点とする辺の集合
  - $\delta^- : V \to 2^E$: 頂点を終点とする辺の集合

$$V=\{v_0,v_1,v_2,v_3\}$$

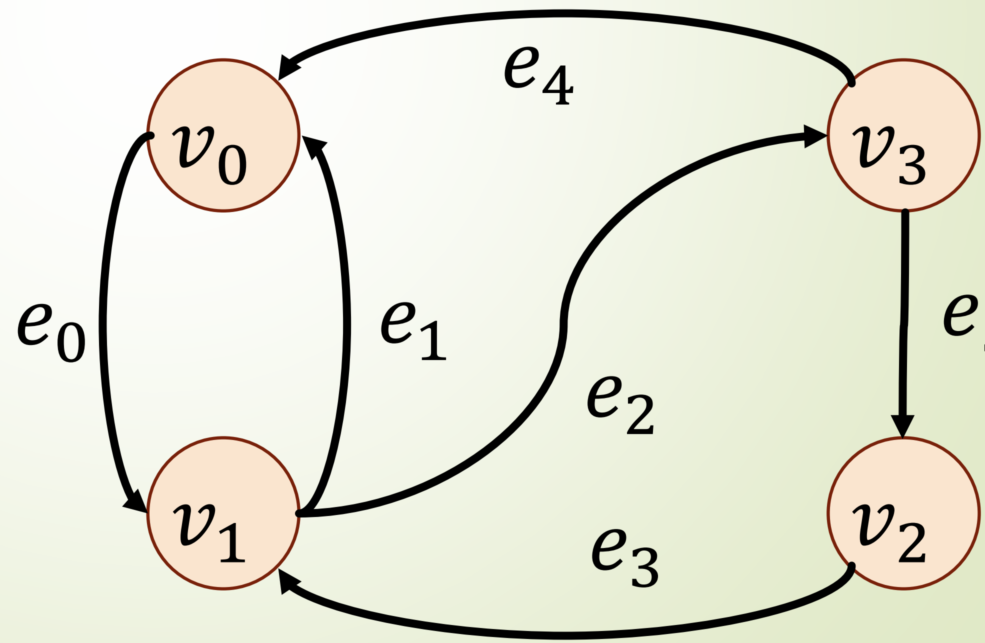
$$E=\{e_0,e_1,e_2,e_3.e_4,e_5\}$$

$$\delta^+v_0=\{e_0\} \qquad \delta^-v_0=\{e_1,e_4\}$$

$$\delta^+v_1=\{e_1,e_2\} \qquad \delta^-v_1=\{e_0,e_3\}$$

$$\delta^+v_2=\{e_3\} \qquad \delta^-v_2=\{e_5\}$$

$$\delta^+v_3=\{e_4,e_5\} \qquad \delta^-v_3=\{e_2\}$$

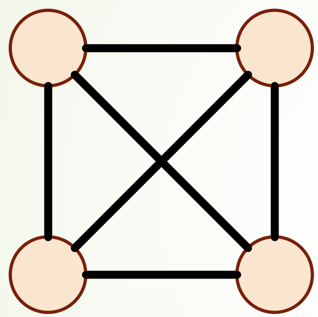


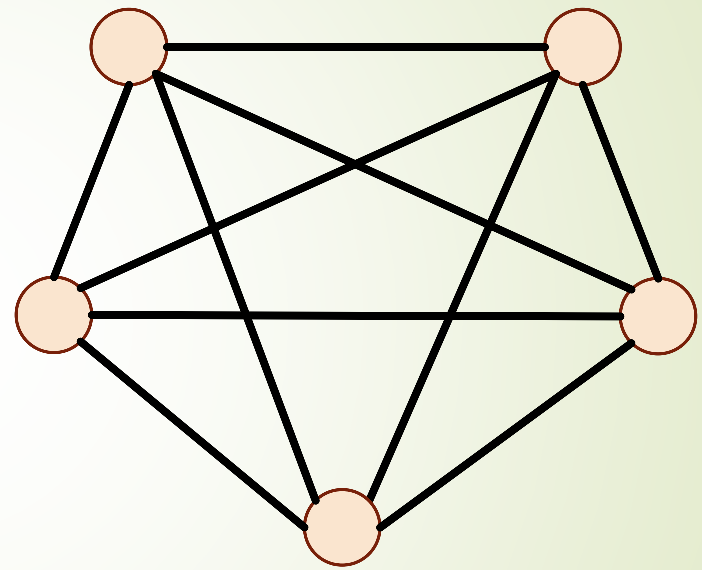
次数(degree)

- 無向グラフ
  - 頂点vを始点とする辺: $\delta: V \to 2^E$
  - 頂点vの次数$|\delta v|$
- 有向グラフ
  - 頂点vを始点とする辺: $\delta^+: V \to 2^E$
  - 頂点vを終点とする辺: $\delta^-: V \to 2^E$
  - 頂点vの正次数$|\delta^+ v|$、負次数$|\delta^- v|$

完全グラフ
Complete Graphs

K_4

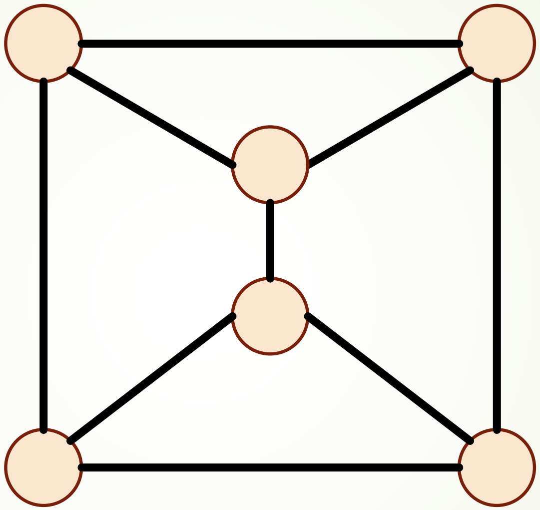




K_5

完全グラフ (Complete Graph)
全ての頂点の組に辺が存在

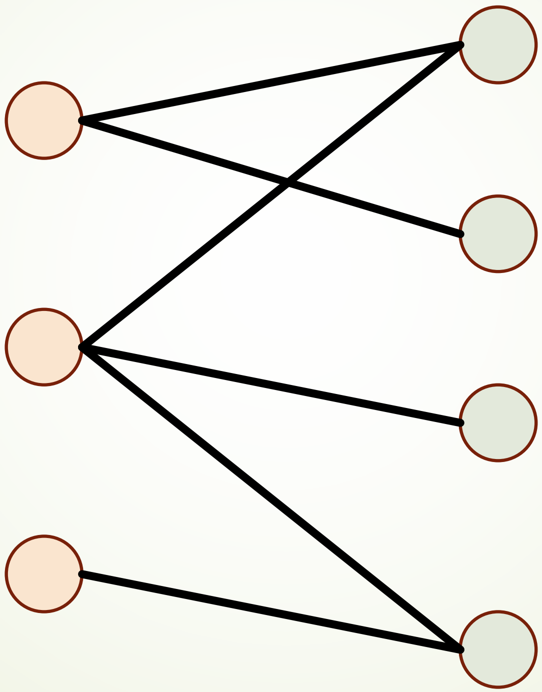
正則グラフ
Regular Graphs



正則グラフ (Regular Graph)
全ての頂点の次数が等しい

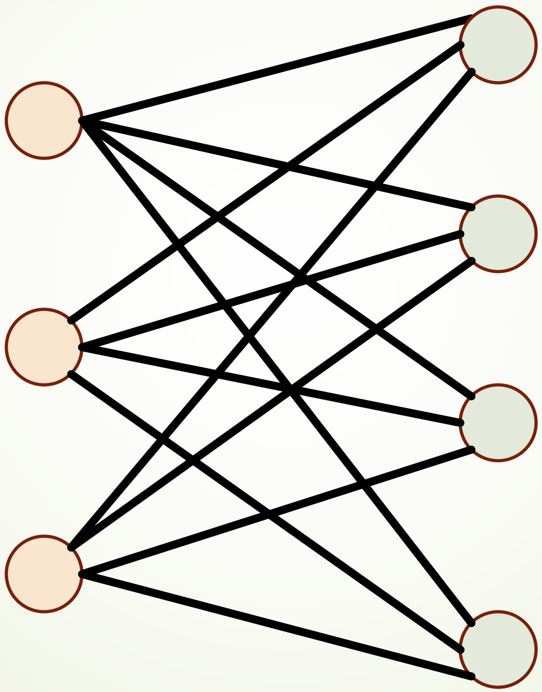
二部グラフ
Bipartite

頂点が二つの集合に分かれている
集合内の辺が無い



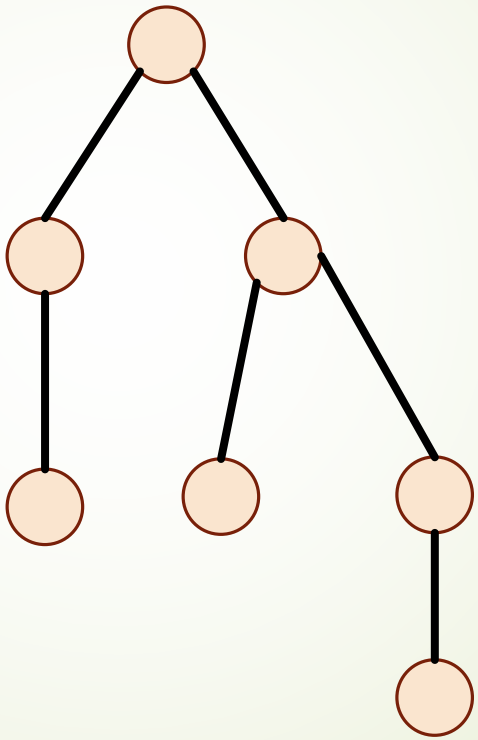
https://oracleofbacon.org/

完全二部グラフ



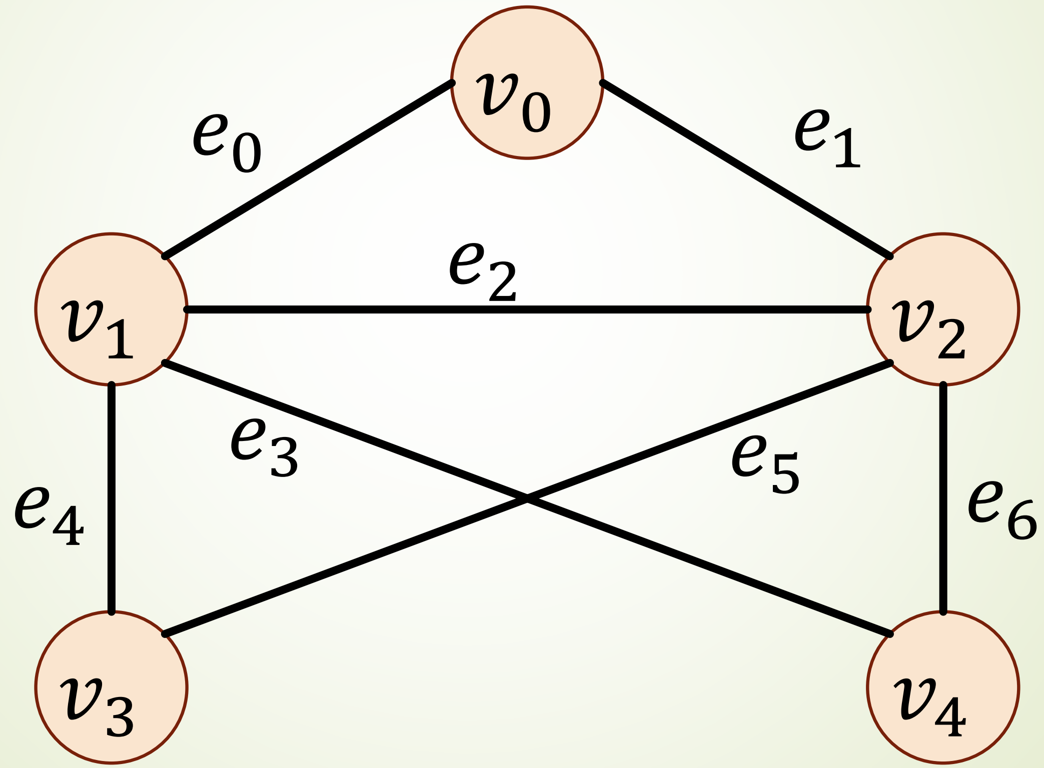
木
Tree

閉路の無いグラフ
有向と無向がある



Euler閉路 (一筆書き)

- 「Königsbergの橋」: Graph理論の端緒
  - Leonhard Euler (1707-1783)
- 無向グラフに対して、全ての辺を一度ずつ通り、元の頂点に戻る道を見つける
- 全ての頂点の次数が偶数の場合のみ、Euler閉路が存在する

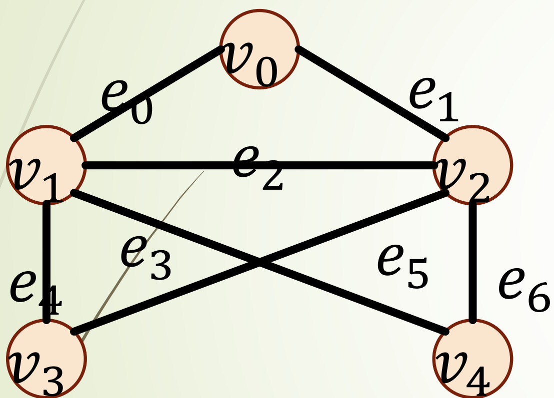


$e_0 \to e_4 \to e_5 \to e_2 \to e_3 \to e_6 \to e_1$

Euler閉路列挙のアルゴリズム

//E_{Euler} : 経由した辺のリスト、初期値$E_{\text{Euler}} = \emptyset$
// r : 始点
enumerateEuler(v, E_{Euler}){
    if (($v == r$) ∧ ($|E_{\text{Euler}}| == |E|$)){
        見つけたEuler閉路E_{Euler}を保存
    } else {
        forall ($e \in \delta v$){//vに接続する全ての辺
            if ($e \notin E_{\text{Euler}}$) {
                $E'_{\text{Euler}} = E_{\text{Euler}} \cup \{e\}$
                $w = \partial e \setminus \{v\}$//辺$e$の$v$と反対の端点
                enumerateEuler(w, E'_{Euler})
            }
        }
    }
}

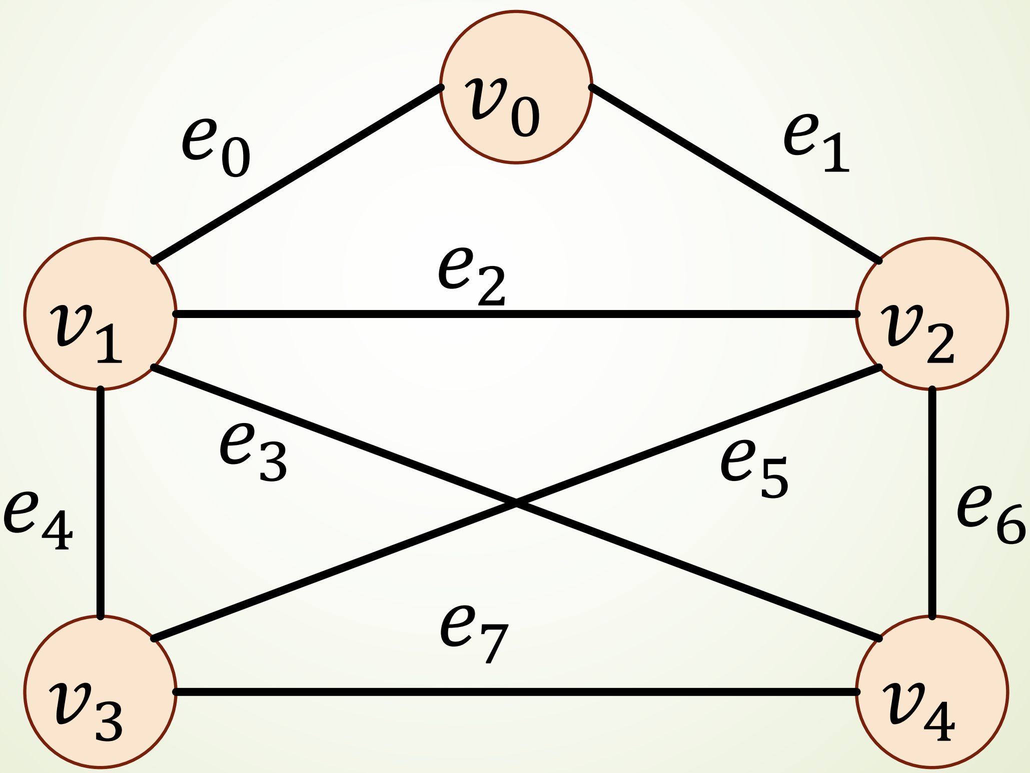
列挙の結果



['e0', 'e2', 'e5', 'e3', 'e4', 'e6', 'e1']
['e0', 'e2', 'e6', 'e4', 'e3', 'e5', 'e1']
['e0', 'e3', 'e5', 'e2', 'e4', 'e6', 'e1']
['e0', 'e3', 'e5', 'e6', 'e4', 'e2', 'e1']
['e0', 'e4', 'e6', 'e2', 'e3', 'e5', 'e1']
['e0', 'e4', 'e6', 'e5', 'e3', 'e2', 'e1']
['e1', 'e2', 'e3', 'e5', 'e6', 'e4', 'e0']
['e1', 'e2', 'e4', 'e6', 'e5', 'e3', 'e0']
['e1', 'e5', 'e3', 'e2', 'e6', 'e4', 'e0']
['e1', 'e5', 'e3', 'e4', 'e6', 'e2', 'e0']
['e1', 'e6', 'e4', 'e2', 'e5', 'e3', 'e0']
['e1', 'e6', 'e4', 'e3', 'e5', 'e2', 'e0']

Hamilton閉路

- 無向グラフに対して、全ての頂点を一度ずつ経由して、始点に戻る閉路
- 巡回セールスマン問題等で必要となる



$$v_0 \to v_1 \to v_4 \to v_3 \to v_2 \to v_0$$

Hamilton閉路列挙のアルゴリズム

//V_{Hamilton} : 経由した頂点のリスト、初期値$V_{\mathrm{Hamilton}} = \{\mathrm{r}\}$

// r : 始点

enumerateHamilton(v, V_{Hamilton}){

    forall ($e \in \delta v$){//vに接続する全ての辺

        $w = \partial e \setminus \{v\}$//辺$e$の$v$と反対の端点

        if ($(w == r) \land (|V_{\mathrm{Hamilton}}| == |V|)$){

            見つけたHamilton閉路V_{Hamilton}を保存

        } else {

            if ($w \notin V_{\mathrm{Hamilton}}$){

                $V'_{\mathrm{Hamilton}} = V_{\mathrm{Hamilton}} \cup \{w\}$

                enumerateHamilton($w, V'_{\mathrm{Hamilton}}$)

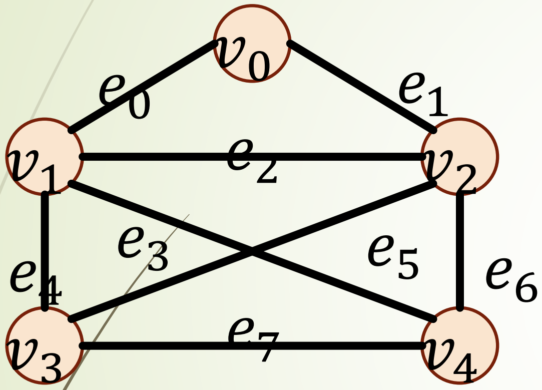
            }

        }

    }

}

列挙の結果



['v0', 'v1', 'v3', 'v4', 'v2']
['v0', 'v1', 'v4', 'v3', 'v2']
['v0', 'v2', 'v3', 'v4', 'v1']
['v0', 'v2', 'v4', 'v3', 'v1']