

1238 計算の理論

上原 隆平

2019年I-1期(4-5月)

I238 Computation Theory

by

Prof. Ryuhei Uehara

Term I-1, April-May, 2019

計算量の理論

- ゴール1:

- “計算可能な関数/問題/言語/集合”

- 関数には2種類存在する;

1. 計算不能(!)な関数
2. 計算可能な関数

- ゴール2:

- 「問題の困難さ」を示す方法を学ぶ

- 計算可能な問題であっても、手におえない場合がある！
 - 計算に必要な資源(時間・領域)が多すぎるとき

Computational Complexity

- Goal 1:
 - “*Computable Function/Problem/Language/Set*”
 - We have two functions;
 1. Functions that are not computable!
 2. Functions that are computable.
- Goal 2:
 - How can you show “*Difficulty of Problem*”
 - There are *intractable* problems even if they are computable!
 - because they require too many resources (time/space)!

計算量の理論

- ゴール1:

- “計算可能な関数/問題/言語/集合”
 - 関連する専門用語;
計算可能性、対角線論法

グラフ理論/グラフの問題/グラフアルゴリズム

- グラフ理論超入門
 - 無向グラフ, 有向グラフ, 多重グラフ, 木構造、
平面グラフ
- グラフ上の問題とアルゴリズム
 - 探索問題: オイラー閉路、最小全域木など

- ゴール2:

- 「問題の困難さ」を示す方法を学ぶ
 - 関連する専門用語;
クラスNP, $P \neq NP$ 予想, NP困難性, 多項式時間還元

Computational Complexity

- Goal 1:

- “*Computable Function/Problem/Language/Set*”
 - Technical terms;
computability, diagonalization

- Goal 2:

- How can you show “*Difficulty of Problem*”
 - Technical terms;

The class NP, $P \neq NP$ conjecture, NP-hardness, polynomial time reduction

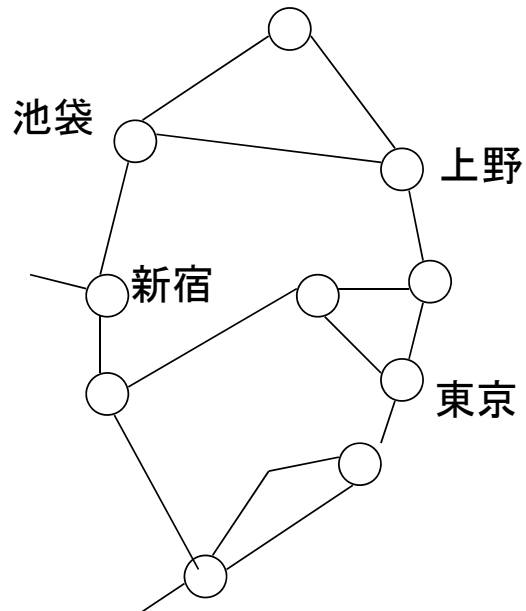
Graph Theory/Graph Problems/ Graph Algorithms

- Introduction to Graph Theory
 - (Un)directed graph, multi-graph, tree, planar graph
- Graph problems and graph algorithms
 - Search Problems: Euler cycle, minimum spanning tree

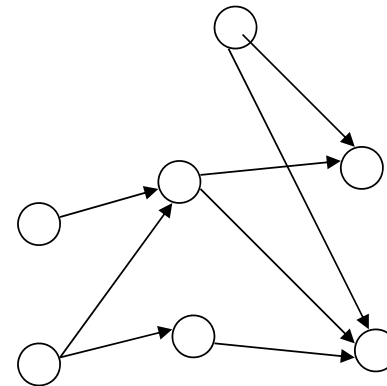
グラフ超入門

- 頂点を辺で結んだもの
 - 無向グラフ: 辺に方向がない
 - 有向グラフ: 辺に方向がある

例: 路線図



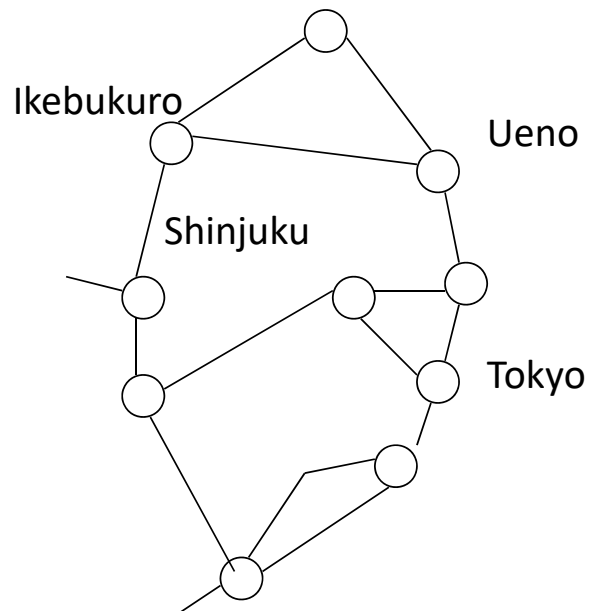
例: 講義の履修順序



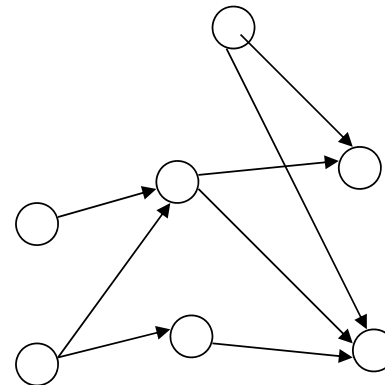
Super Introduction to Graph Theory

- A graph consists of a set of **vertices** joined by **edges**
 - (Undirected) graph: each edge has no direction
 - Directed graph: an edge has a direction

Ex: Railways



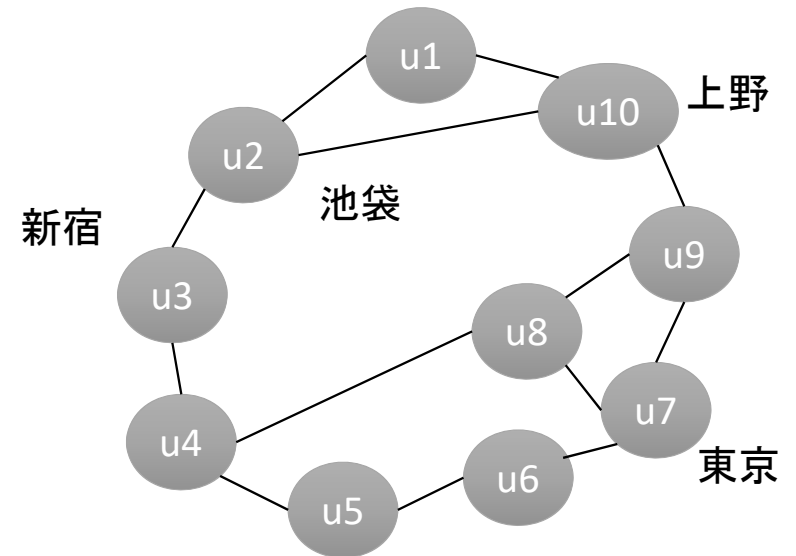
Ex: Relationship between courses



グラフ: 表記

- グラフ: $G = (V, E)$
 - V : 頂点集合, E : 辺集合
- 頂点: $u, v, \dots \in V$
- 辺: $e = \{u, v\} \in E$ (無向辺)
 $a = (u, v) \in E$ (有向辺)
- 頂点や辺は重みを持つことも
 - $w(u), w(e)$
 - 距離, 金額, 時間など
- 頂点の次数
 - 無向グラフの場合
頂点 v につながっている辺の数
 $\text{deg}(v)$ と書く
 - 有向グラフの場合
入次数/出次数を区別する
 $\text{indeg}(v), \text{outdeg}(v)$ などと書く

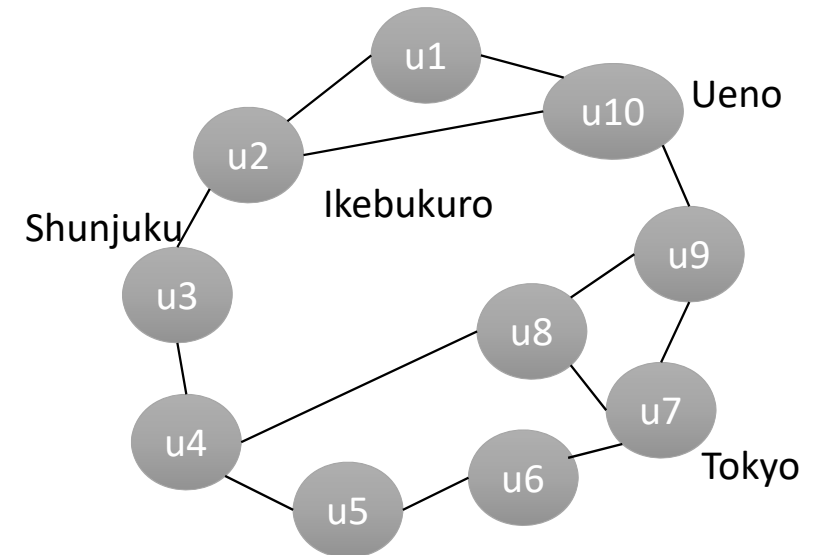
なぜか $|V|=n, |E|=m$
と書く「常識」がある



Graph: Notation

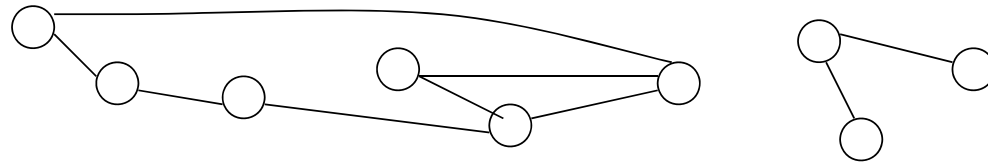
- Graph: $G = (V, E)$
 - V : Vertex set, E : Edge set
- Vertex: $u, v, \dots \in V$
- Edge: $e = \{u, v\} \in E$ (undirected)
 $a = (u, v) \in E$ (directed)
- Sometimes, vertices and edges may have “weights”
 - $w(u), w(e)$
 - Distance, cost, time, etc.
- Degree of a vertex
 - Undirected graph:
The number of edges incident to a vertex v denoted by $\text{deg}(v)$
 - Directed graph:
Indegree and Outdegree are distinguished denoted by, e.g., $\text{indeg}(v), \text{outdeg}(v)$

It is common to denote by
 $|V|=n, |E|=m.$

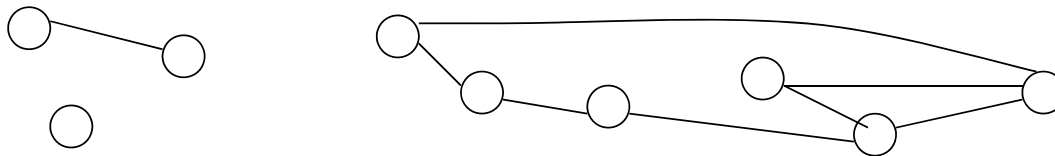


グラフ: 基本的な用語 (1/2)

- 路 (path): 辺で隣り合う頂点を繋いだもの
 - 単純路 (simple path): 同じ頂点を複数回通らない

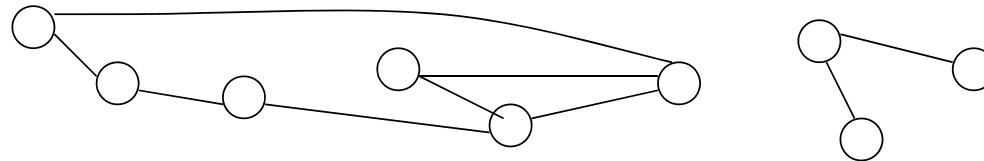


- 閉路 (cycle, closed path): v から v への路
- 連結グラフ (connected graph): どの2頂点間にも路が存在するグラフ

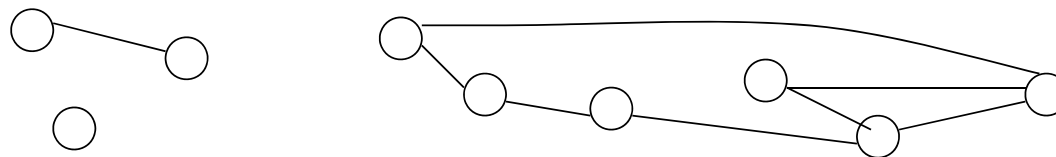


Graph: Basic terms (1/2)

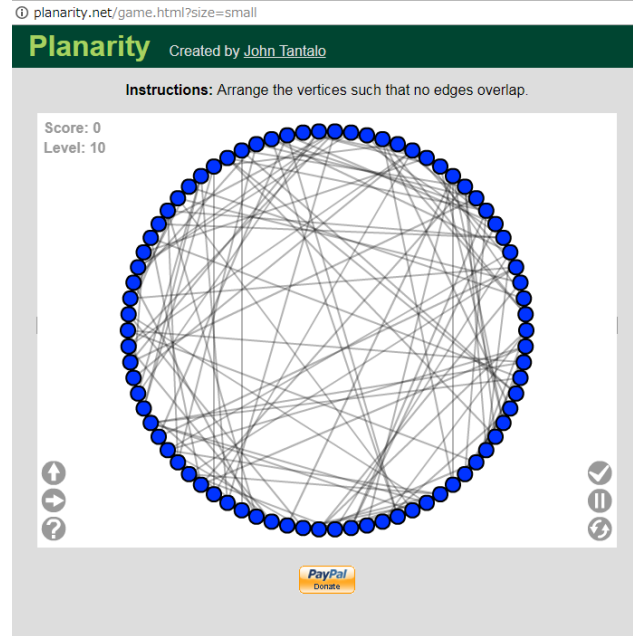
- Path: a sequence of vertices joined by edges
 - Simple path: It never visit the same vertex twice or more



- Cycle, closed path: a simple path from v to v
- Connected graph: graph s.t. any two vertices are joined by a path

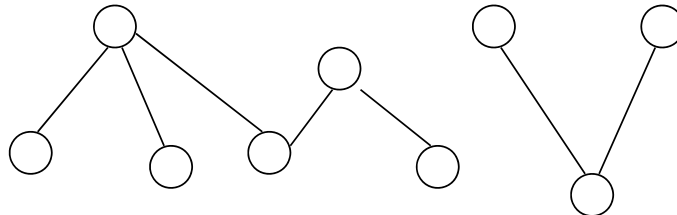


<http://planarity.net/> より
(かなりハマるので危険)



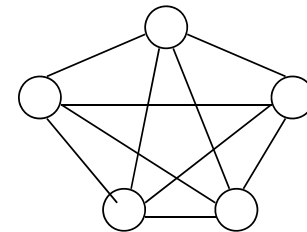
グラフ: 基本的な用語 (2/2)

- 森 (forest): 閉路を含まないグラフ
- 木 (tree): 連結で閉路を含まないグラフ



- 平面的グラフ (planar graph): 辺の交差なしで平面に描画できるグラフ
- (平面グラフ(plane graph): 描画情報付きの平面グラフ)
- 完全グラフ (complete graph): 全ての頂点对を辺で隣接させたグラフ
 - 完全グラフ K_5 は極小非平面的グラフ

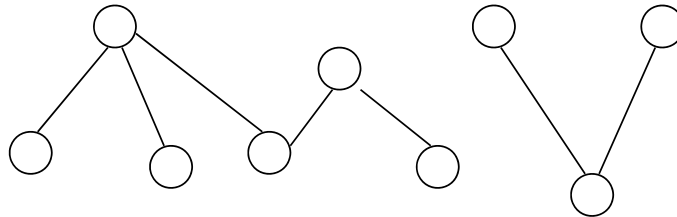
(証明は難しい: やって見たけどできなかったでは証明にならない)



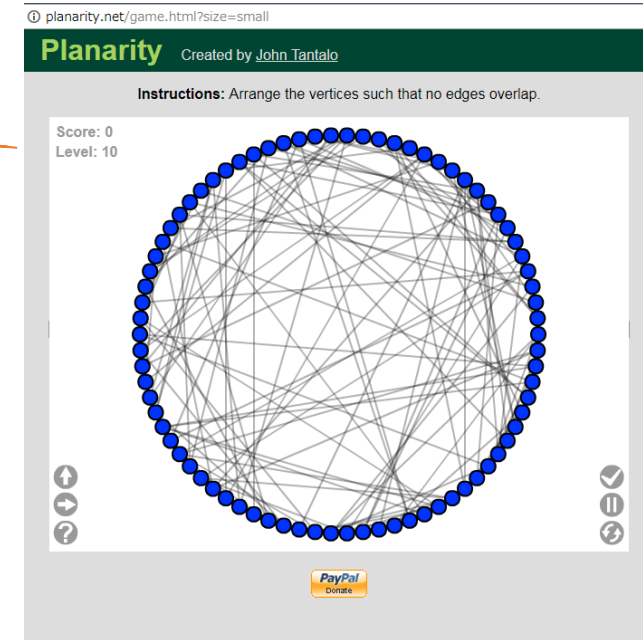
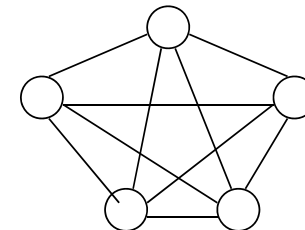
<http://planarity.net/>
(It is dangerous for students 😊)

Graph: Basic terms (2/2)

- Forest: graph having no cycle (or acyclic)
- Tree: connected acyclic graph



- Planar graph: a graph drawable with no crossing of edges
- (Plane graph: Planar graph with plane drawing)
- Complete graph: graph s.t. all pairs are joined by edges
 - A complete graph K_5 is a minimal non-planar graph



グラフの基本的な性質

定理(握手補題): 任意の無向グラフ $G=(V,E)$ について,

$$\sum \deg(v) = 2|E|$$

[証明] それぞれの辺は, 両端点で, 上記の式の左辺に2ずつ貢献するから.

定理: 頂点数 n の木の辺の数は $n-1$

[証明] 略.

定理: 頂点数 n の平面グラフの辺の数は高々 $3n-6$

[証明] 略.

練習問題:
握手補題の有向
グラフ版を考えよ

練習問題:
木には葉(次数が1の頂点)が
必ず存在することを証明せよ

Basic properties of graphs

Exercise:
Consider directed
version

Theorem (Handshake lemma): For any undirected graph $G=(V,E)$,

$$\sum \deg(v) = 2|E|$$

[Proof] Each edge contributes twice to the left hand at its both endpoints.

Theorem: The number of edges of a tree of n vertices is $n-1$

[Proof] Omitted.

Theorem: The number of edges of a planar graph of n vertices is at most $3n-6$

[Proof] Omitted.

Exercise:
Prove that there exists at least one leaf
(vertex of degree 1) in any tree.

グラフアルゴリズムの計算量

- 頂点数 n , 辺数 $m \in O(n^2)$
 - 無向グラフ: $m \leq n(n-1)/2$
 - 有向グラフ: $m \leq n(n-1)$
- 平面グラフでは $m \in O(n)$
- グラフアルゴリズムの計算量は n や m の式で表す

Computational complexity of graph algorithms

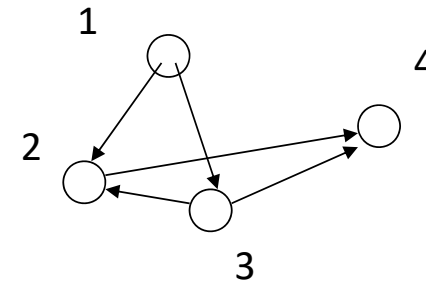
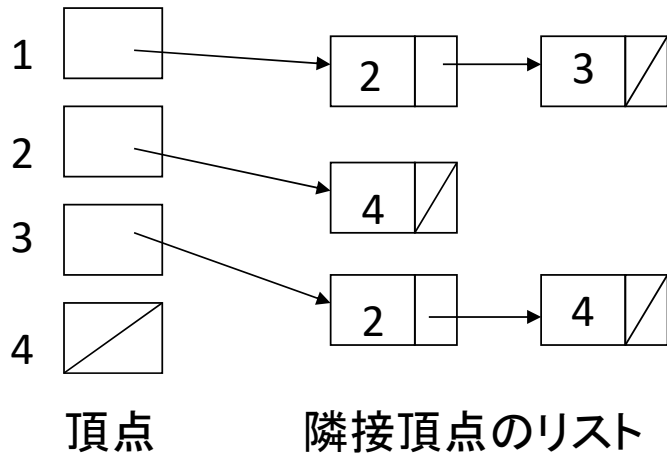
- n vertices, m edges, $m \in O(n^2)$
 - Undirected graph: $m \leq n(n-1)/2$
 - Directed graph: $m \leq n(n-1)$
- On a planar graph, $m \in O(n)$
- Computational complexity of a graph algorithm is described by a function of n and m .

グラフの表現方法

- 隣接行列

$$M = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

- 隣接リスト

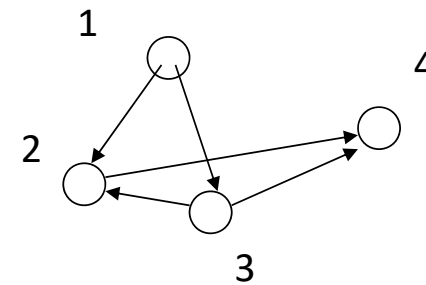
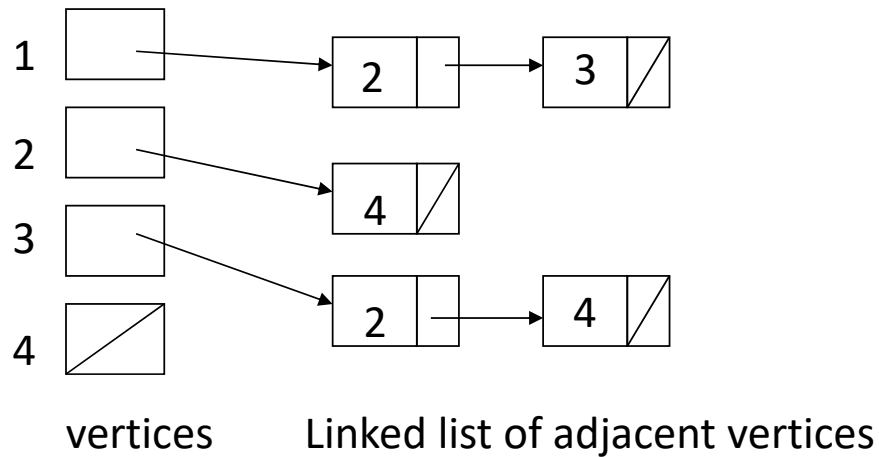


Representations of a graph

- Adjacency matrix

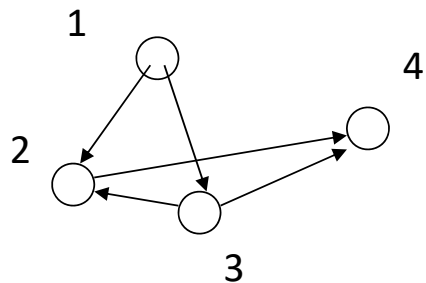
$$M = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

- Adjacency list



グラフの表現方法: 行列表現(隣接行列)

- $(u, v) \in E \Rightarrow M[u, v] = 1$
- $(u, v) \notin E \Rightarrow M[u, v] = 0$



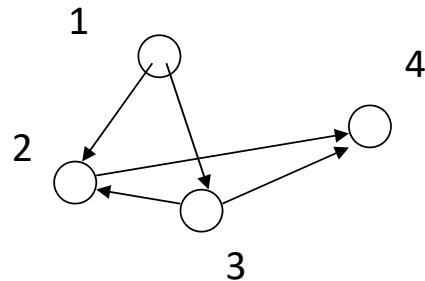
$$M = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

辺重み付きグラフ
に拡張するのは
簡単

Representation of a graph: Matrix representation (adjacency matrix)

- $(u, v) \in E \Rightarrow M[u, v] = 1$
- $(u, v) \notin E \Rightarrow M[u, v] = 0$

It is easy to
extend to edge-
weighted graph.

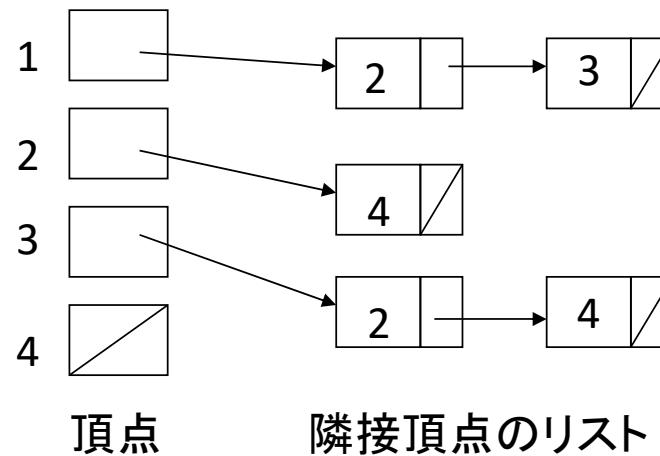
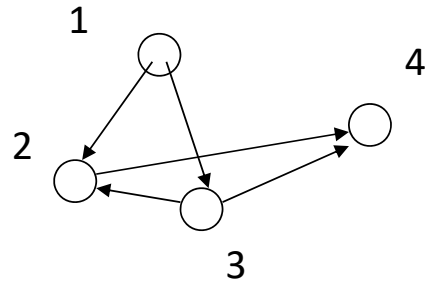


$$M = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

グラフの表現方法: リスト表現(隣接リスト)

- $(u, v) \in E \Leftrightarrow v \in T(u)$
 - $T(u)$ は u の隣接点のリスト

• 例:

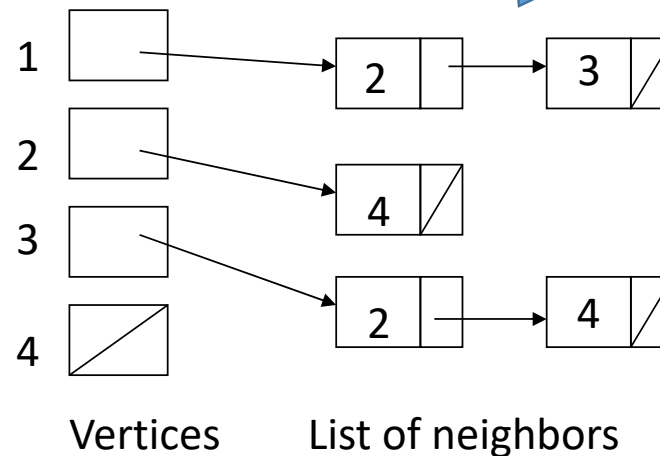
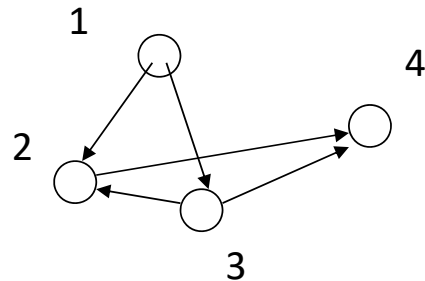


頂点重み付きグラフ
に拡張するのは簡単

Representation of a graph: List representation (adjacency list)

- $(u, v) \in E \Leftrightarrow v \in T(u)$
 - $T(u)$ is the list of neighbors of u

- Example:



It is easy to extend to vertex-weighted graph.

隣接行列 vs 隣接リスト

- メモリ量
 - 隣接行列: $\Theta(n^2)$
 - 隣接リスト: $\Theta(m \log n)$
- 隣接チェックにかかる時間: $(u, v) \in E$?
 - 隣接行列: $\Theta(1)$
 - 隣接リスト: $\Theta(n)$

Q. グラフの更新 (e.g., 頂点・辺の追加・削除) は？

Adjacency matrix v.s. Adjacency list

- Memory
 - Adjacency matrix: $\Theta(n^2)$
 - Adjacency list: $\Theta(m \log n)$
- Time to check if $(u, v) \in E$?
 - Adjacency matrix: $\Theta(1)$
 - Adjacency list: $\Theta(n)$

**Q. How about update of the graph?
(e.g., Add/remove of a vertex/edge)**

ミニ演習 (Some exercises)

- 頂点数4の無向グラフを全て描け(Draw all undirected graphs of 4 vertices)
 - 頂点にラベルはないものとする(Each vertex has no label)
 - 描き方を変えれば同じになるものは省く(“Same” graphs should be reduced)
 - cf. グラフの同型性(Graph Isomorphism)
- 頂点数 n の無向グラフは高々 2^{n^2} 個しかないことを証明せよ
(Prove the number of undirected graphs of n vertices is at most 2^{n^2} .)