グラフ

グラフの表現法・グラフの探索

世の中はネットワークでできている

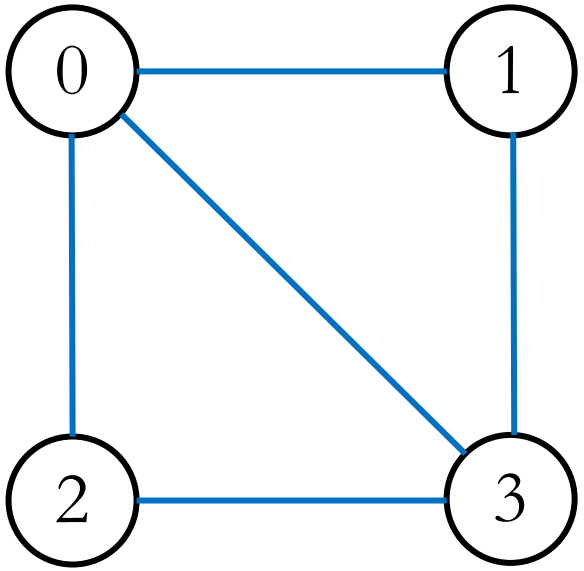
- 航空路や道路・鉄道などの交通網、インターネットなどの物理的なネットワークから、人間関係・会社の取引関係・ソーシャルネットワークなど人と人のつながりの社会的ネットワークまで、今の世の中はネットワークでできている

ネットワークはグラフで表現できる

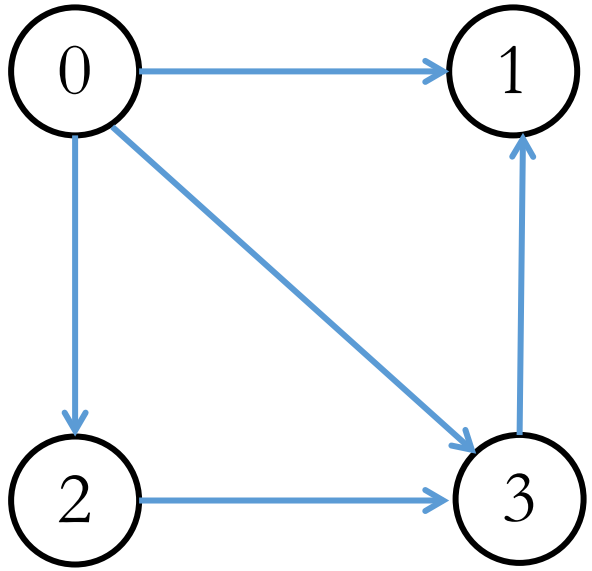
- このような実世界を情報科学的に表現するために、抽象化・モデル化する必要がある
- それに用いるのがグラフ。言い換えると、グラフがそれらのモデルとして利用される

グラフ（Graph）

- 頂点（vertex, node, 節点）の集合Vと辺（edge, link）の集合Eからなる。G=(V, E)と表す。また、|V|で頂点の数、|E|で辺の数を表す

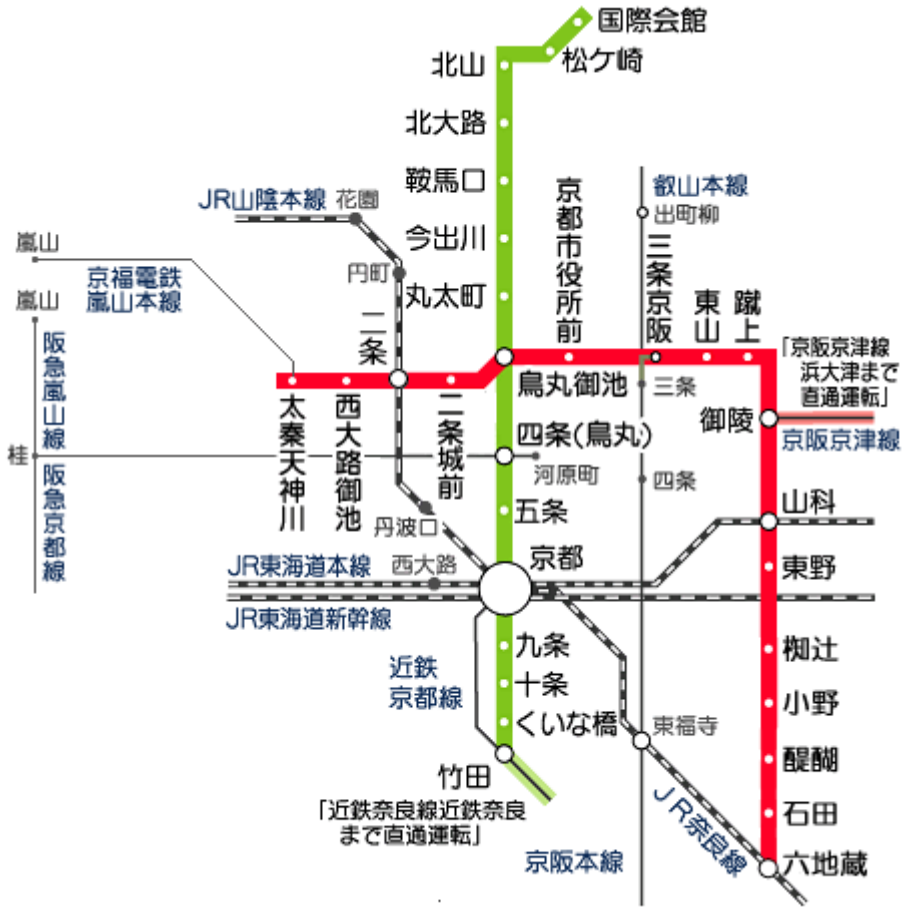


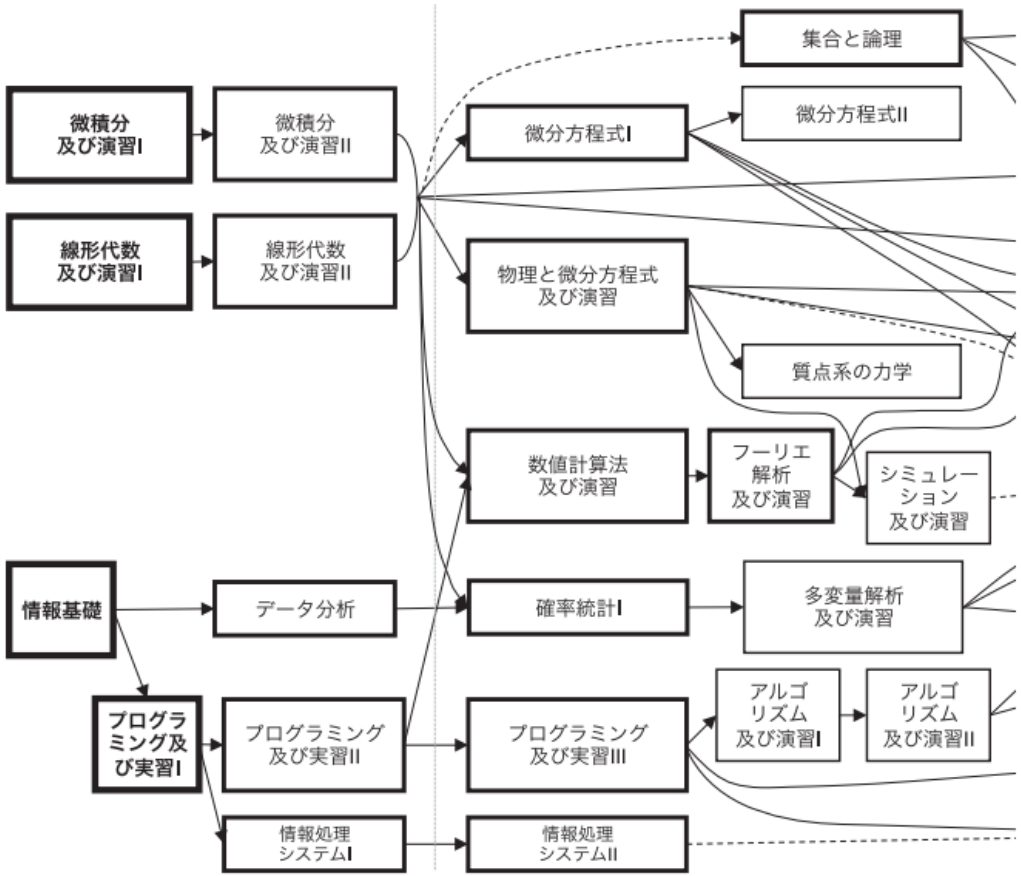
無向グラフ
（例：路線図）



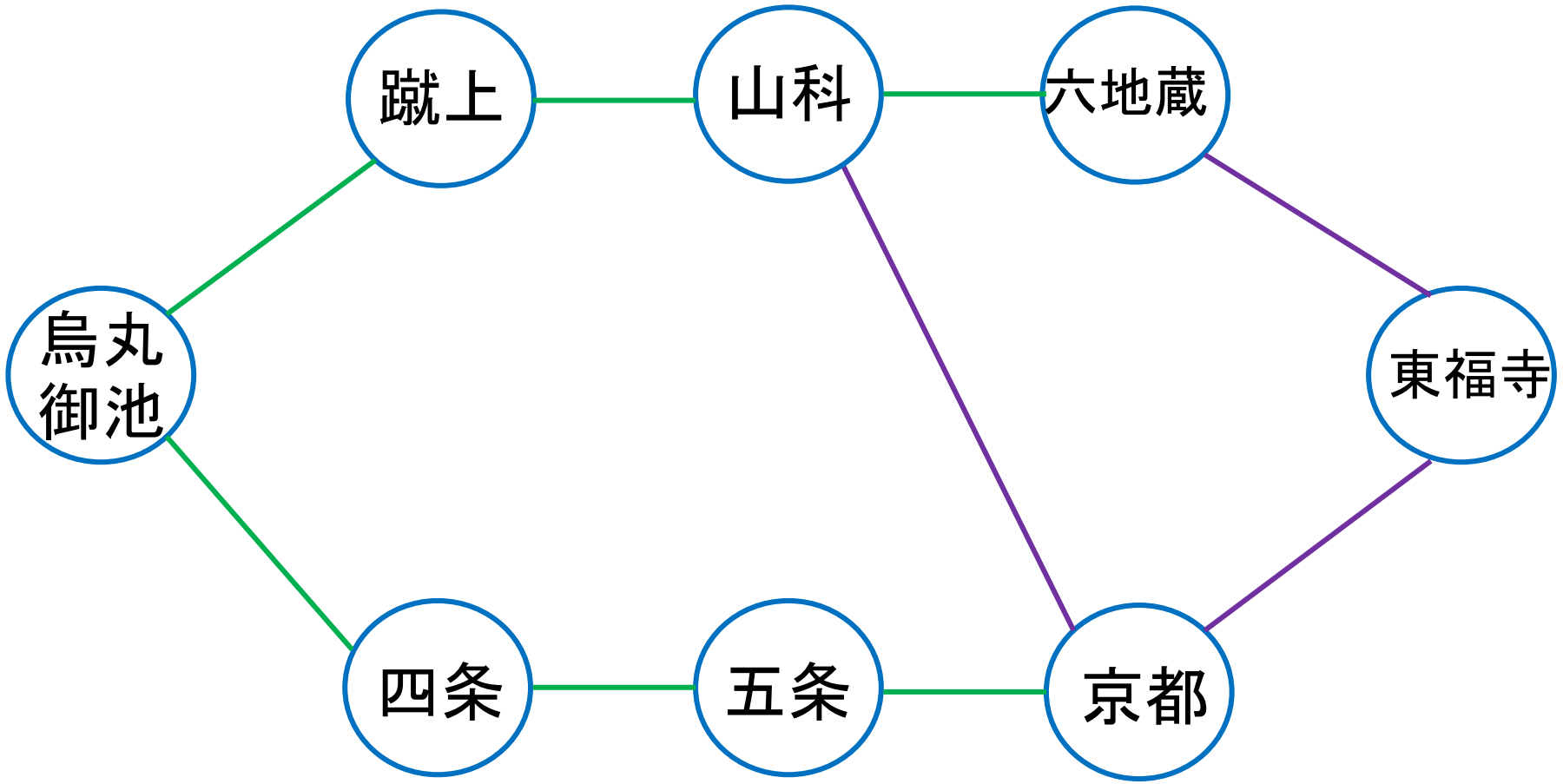
有向グラフ
（例：講義の履修順序）

無向グラフの例・有向グラフの例





路線図をグラフで表現すると



ネットワーク以外にもさまざまな実世界をグラフで表現できる

- 都市巡回
  - 頂点：都市、辺：航空路や鉄道
- 履修科目の依存関係（履修順序）
  - 頂点：科目、辺：科目間順序・依存
- 日本語文の係り受け（依存）関係
  - 頂点：文節、辺：依存関係
- WWW (World Wide Web)
  - 頂点：Webページ、辺：リンク
- 神経回路や分子構造
  - 頂点：ニューロン（神経細胞）、辺：シナプス（結合）
  - 頂点：原子、辺：結合
- 論理回路
  - 頂点：論理ゲート　辺：配線

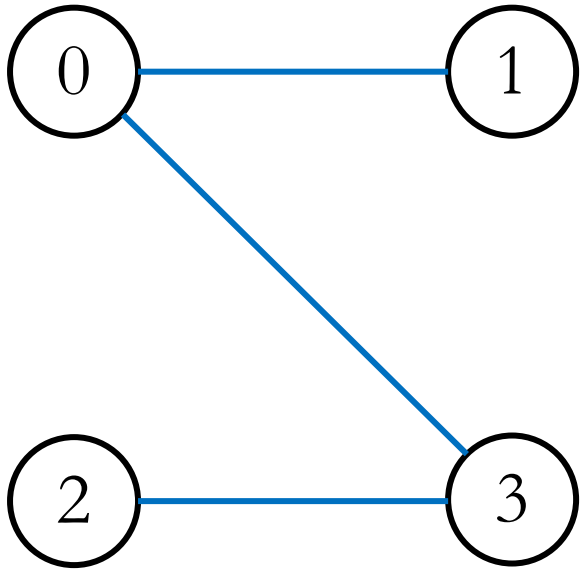
グラフの主な用語

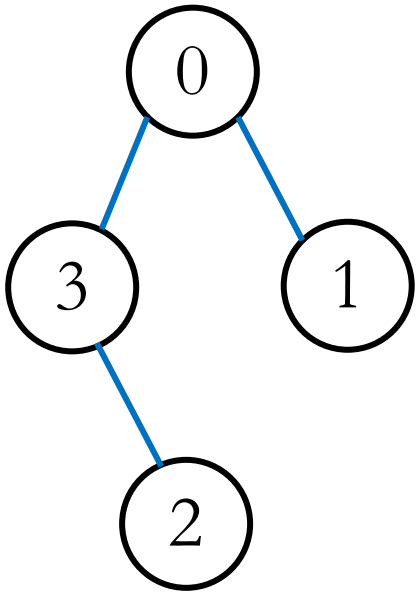
- 有向グラフ・無向グラフ：辺に向きのあるなし
- 重み：辺の属性。時間、距離、コストなど
- 重み付きグラフ：重みのあるグラフ
- 隣接・接続：頂点と頂点が辺でつながっていることを「隣接」しているという。頂点と辺がつながっていることを「接続」という
- 次数（degree）・最大次数：頂点vにつながっている辺の数をこの頂点の次数という。deg(v)と表記する。最多（最少）の辺につながっている頂点の次数を最大（最少）次数という。$\Delta(G)$, $\delta(G)$と表記する

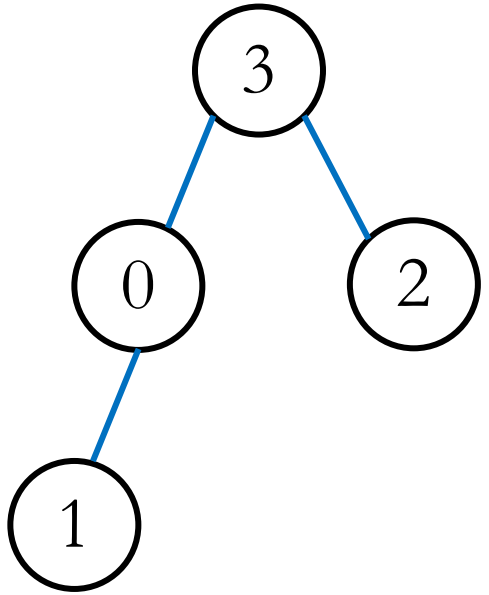
グラフの主な用語

- 道（パス）：頂点と頂点が結ぶ経路
- 閉路：頂点からその頂点自身への道が存在するとき、この道を閉路という
- 連結と非連結：連結とは任意の2頂点間を行き来できること。行き来できない頂点が存在すれば非連結という
- 連結グラフ：連結なグラフ
- 密なグラフ・疎なグラフ：辺の数の多いグラフ、少ないグラフ

木はグラフの一種である

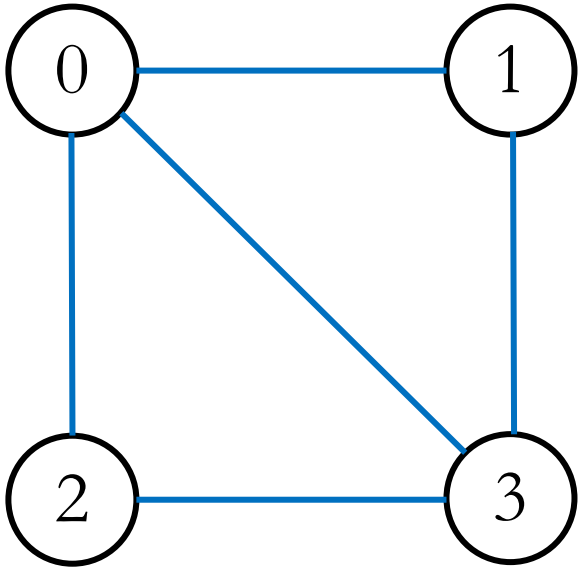






グラフの表現法

隣接行列（Adjacency Matrix）

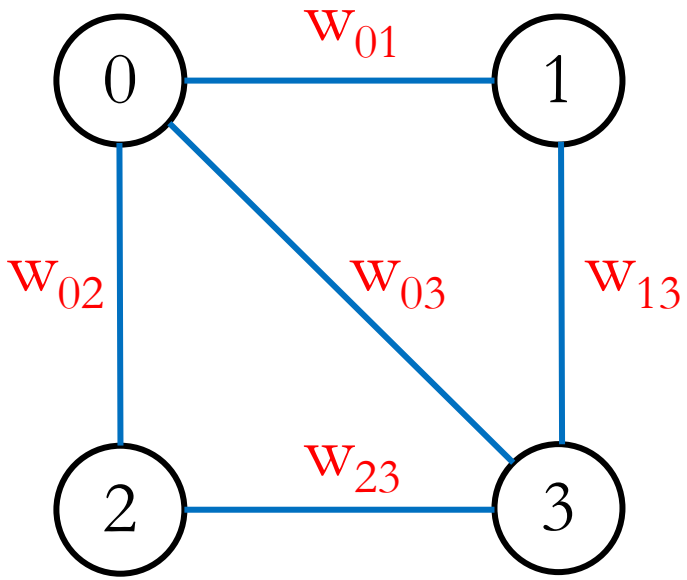


無向グラフ

	0	1	2	3
0	0	1	1	1
1	1	0	0	1
2	1	0	0	1
3	1	1	1	0

対称行列

隣接行列

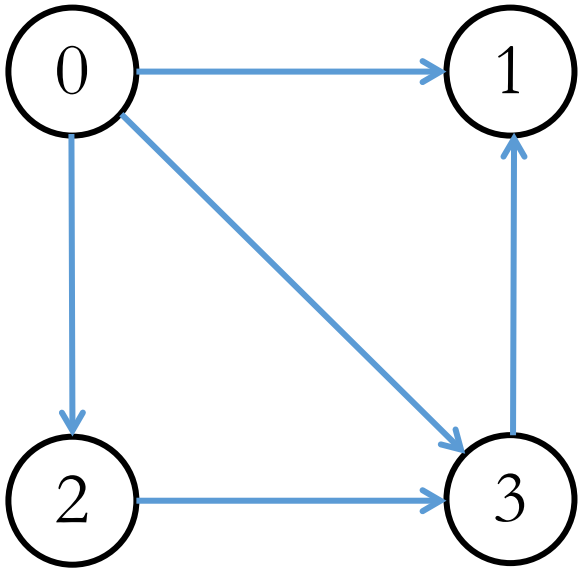


重み付き
無向グラフ

	0	1	2	3
0	0	w_{01}	w_{02}	w_{03}
1	w_{12}	0	0	w_{13}
2	w_{02}	0	0	w_{23}
3	w_{03}	w_{13}	w_{23}	0

対称行列

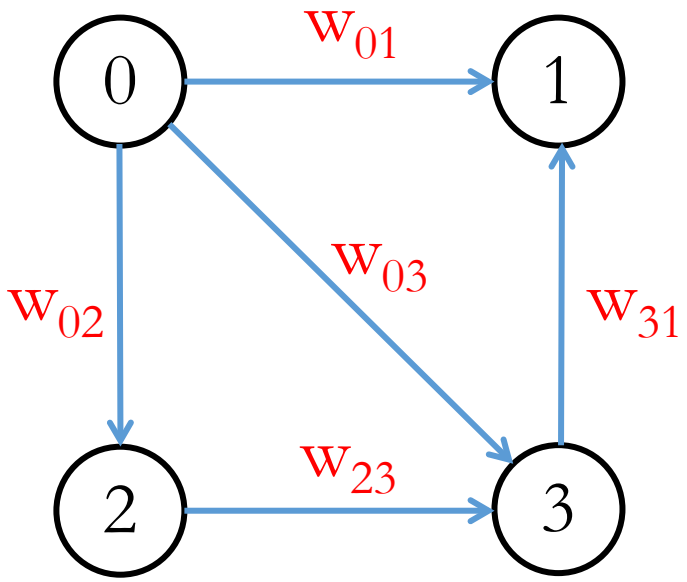
隣接行列



有向グラフ

	0	1	2	3
0	0	1	1	1
1	0	0	0	0
2	0	0	0	1
3	0	1	0	0

隣接行列



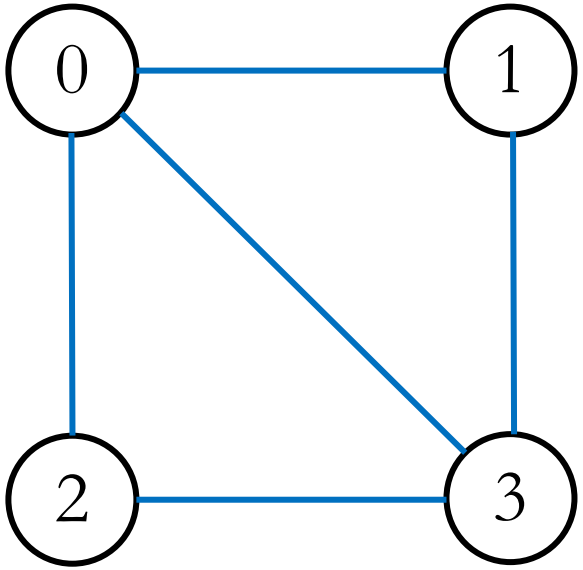
重み付き
有向グラフ

	0	1	2	3
0	0	w_{01}	w_{02}	w_{03}
1	0	0	0	0
2	0	0	0	w_{23}
3	0	w_{31}	0	0

隣接行列

- 頂点（行）と頂点（列）がつながっているかの行列
- 実装が簡単
- 空間（領域）計算量：$O(|V|^2)$
  - どんなグラフでも計算量が同じなので、疎なグラフに対して無駄が多い
- 時間計算量：
  - 2頂点間の隣接関係チェック：$O(1)$
- 辺の重みの表現が簡単

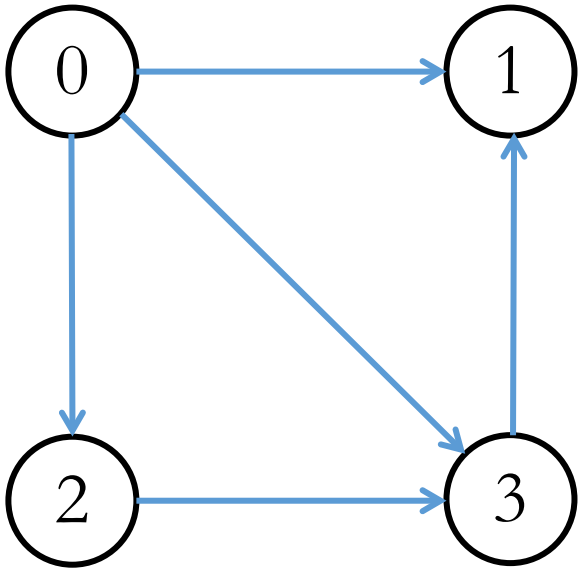
隣接リスト（Adjacency List）



無向グラフ

$0 \rightarrow [1, 2, 3]$
$1 \rightarrow [0, 3]$
$2 \rightarrow [0, 3]$
$3 \rightarrow [0, 1, 2]$

隣接リスト



有向グラフ

$0 \rightarrow [1, 2, 3]$
$1 \rightarrow []$
$2 \rightarrow [3]$
$3 \rightarrow [1]$

隣接リスト

- 各頂点について、出る辺の行き先となる頂点のリスト
- 実装：連結リストやハッシュテーブルなど
- 空間（領域）計算量：　最悪$O(|V|^2)$，平均$O(|V|+|E|)$
  - ほぼフル結合のような密なグラフでは$|E|=O(|V|^2)$なので、上記両者は同一となる
  - $|E|$が$O(|V|)$程度以下の疎なグラフでは計算量が$O(|V|)$となる
- 時間計算量
  - 2頂点間の隣接関係チェック：$O(|V|)$
- 辺の重み：行き先と重みのペアのリストで表現できる

manaba小テスト：03-1

- 10分
- 10点

グラフの探索（Graph Search）

グラフの探索

- グラフのアルゴリズムで最も基本なのは、すべての頂点に何らかの処理を加えて回ること。頂点の訪問ともいう
- そのためには、それぞれの頂点を2回以上回ることなく、しかし必ず1回は回る組織的な（システム的な）手順が必要
- グラフ全体を組織的に調べることをグラフの探索という
- ここでいう探索はグラフの構造を反映した手順
  - たとえば、頂点aから頂点bにたどり着くまでの距離（aからbへの道を形成する辺の重みの総和）。「すべての辺の重みの総和」を求めるとは違う
- 探索方法：
  - 深さ優先探索
  - 幅優先探索

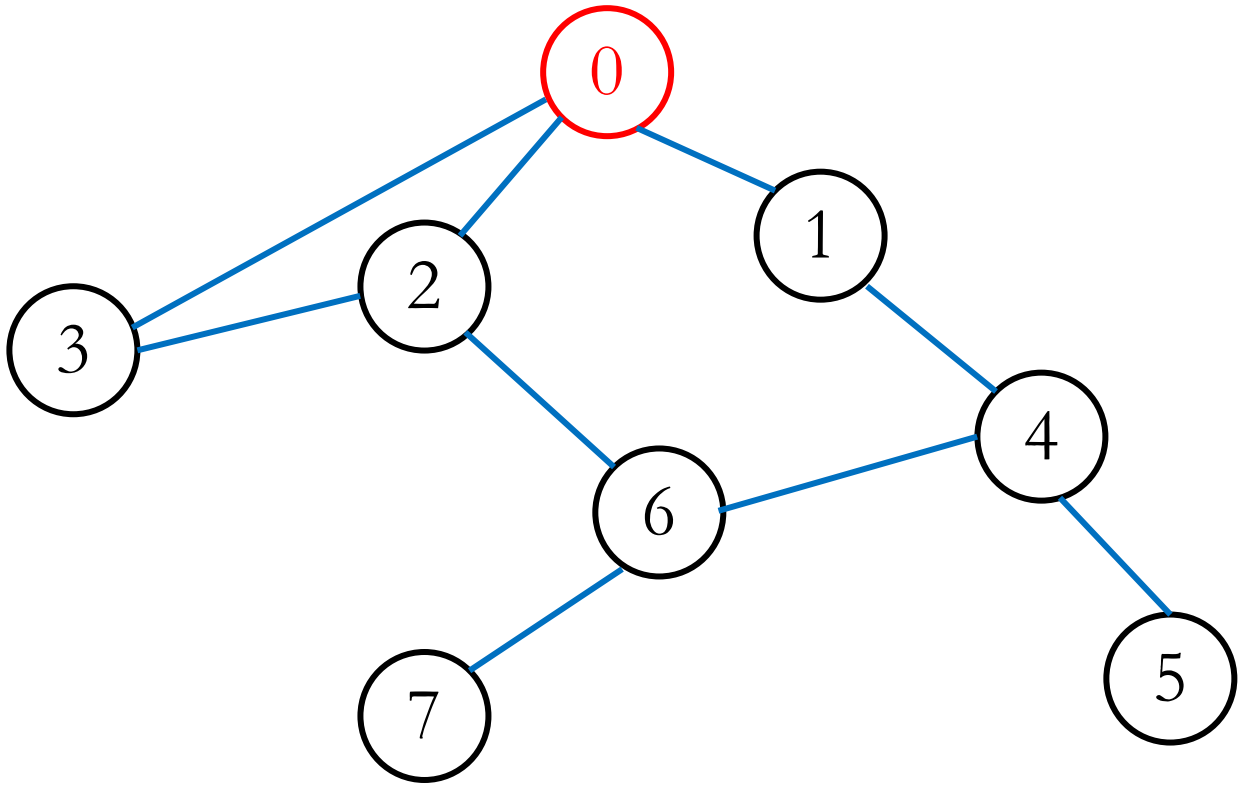
深さ優先探索（DFS: Deep-First Search）

- １つの道を選んで、行けるところまで行ってそれ以上進めなくなったら一歩戻ってそこから新しい道を探す方法。縦型探索とも言われる
- とにかくいけるところまで行って、というのがポイント

幅優先探索（BFS: Breadth-First Search）

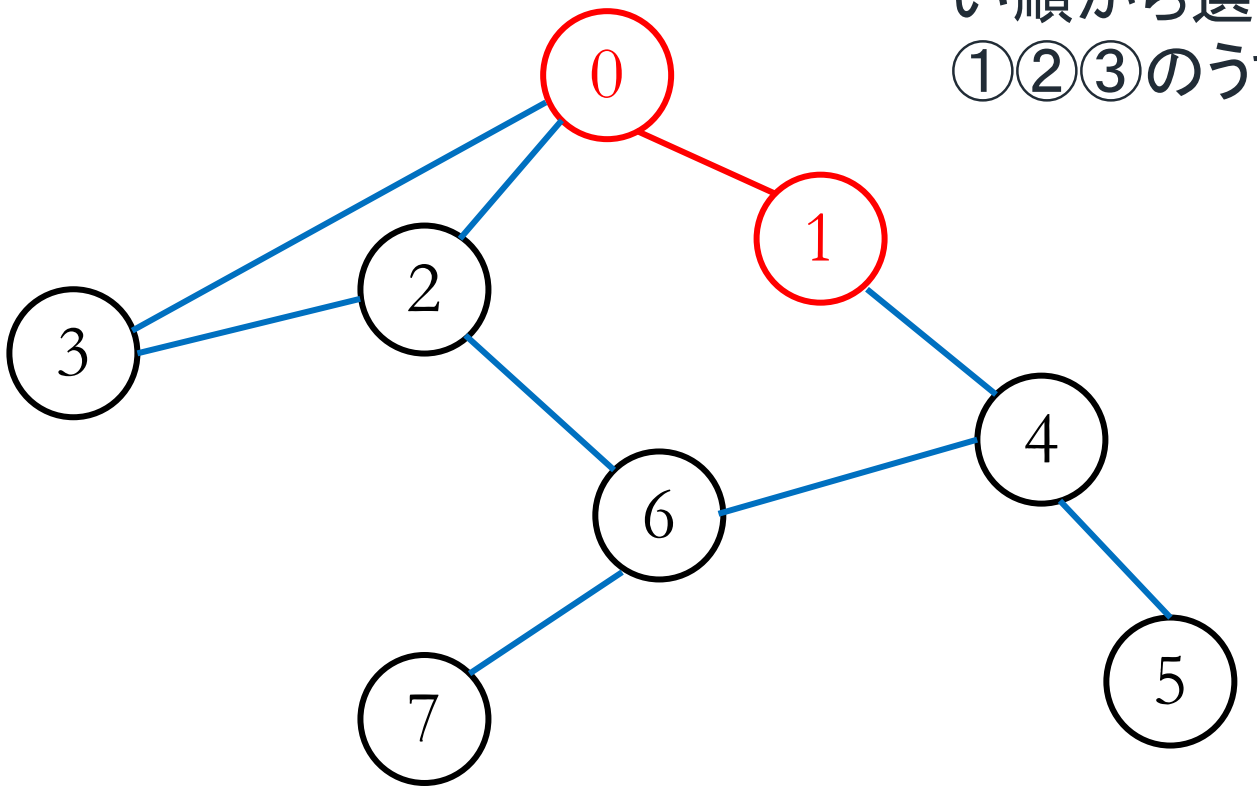
- 最初の頂点を訪問した後、その頂点に隣接しているすべての頂点を順次訪問する。次にそれらと隣接している頂点をさらに順次訪問する。以上を繰り返す方法。横型探索とも言われる
- とにかく出発点に近い点から順に探索する、というのがポイント

深さ優先探索（イメージ）

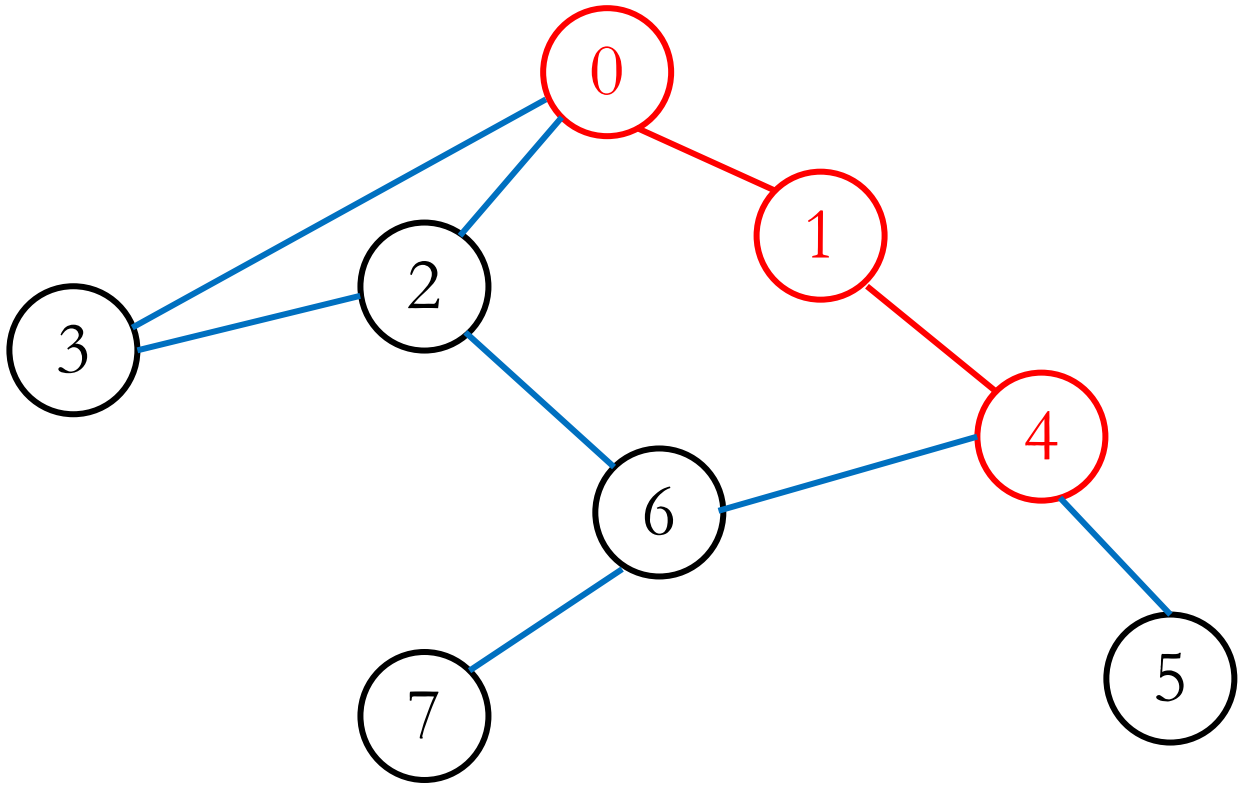


深さ優先探索（イメージ）

複数の選択があるとき、番号の若い順から選択することとする
①②③のうち①を選択

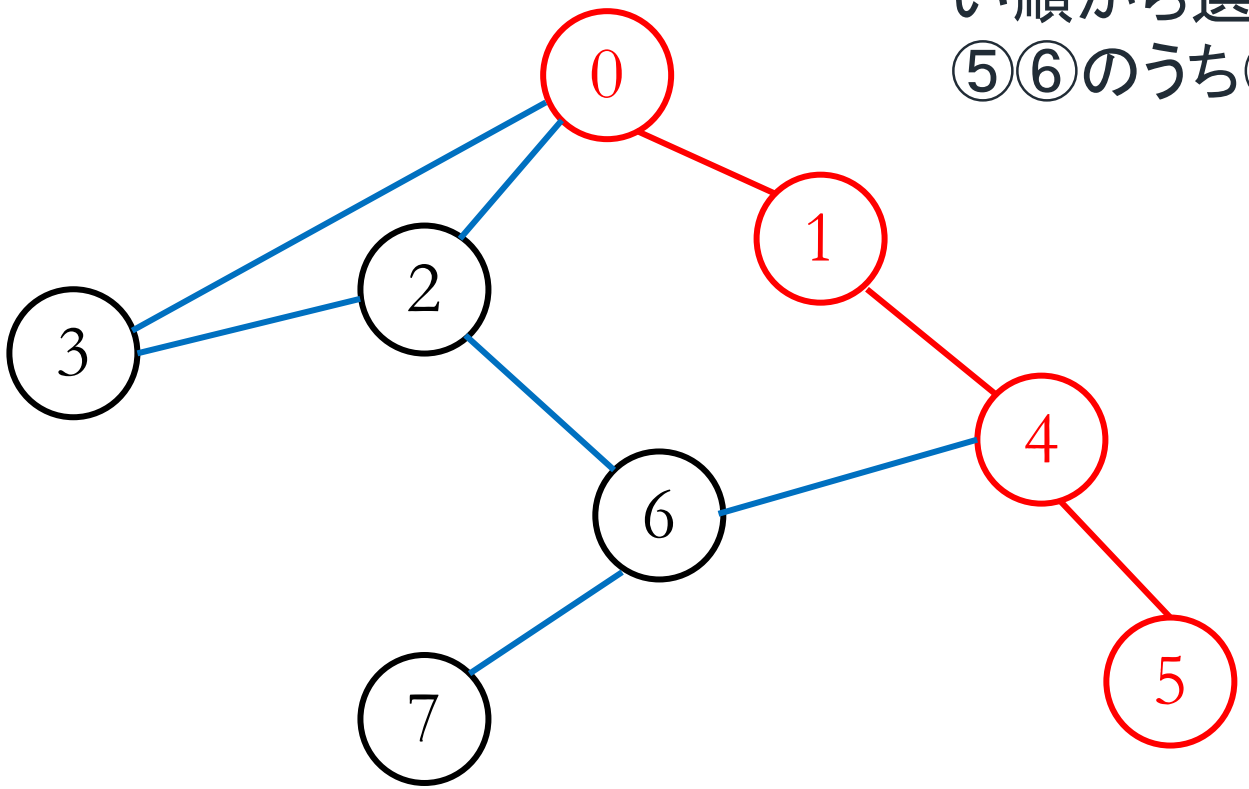


深さ優先探索（イメージ）



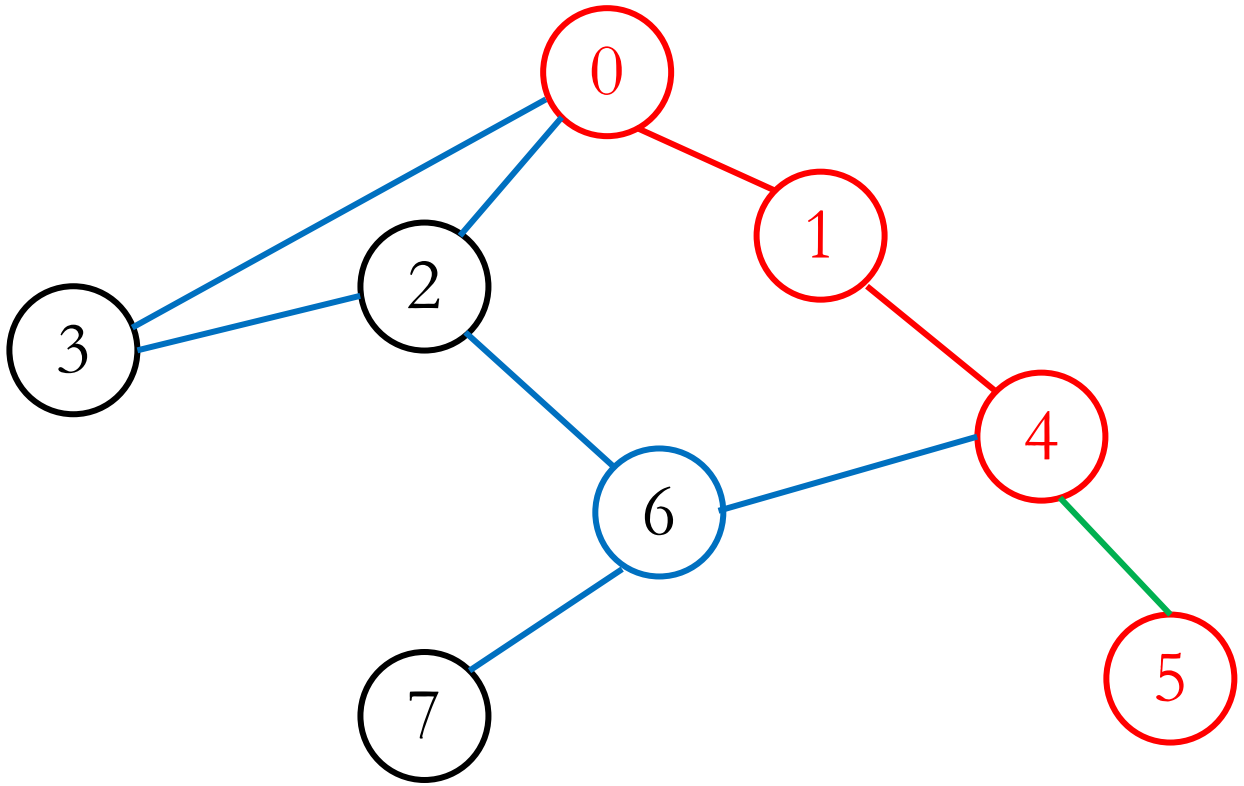
深さ優先探索（イメージ）

複数の選択があるとき、番号の若い順から選択
⑤⑥のうち⑤を選択



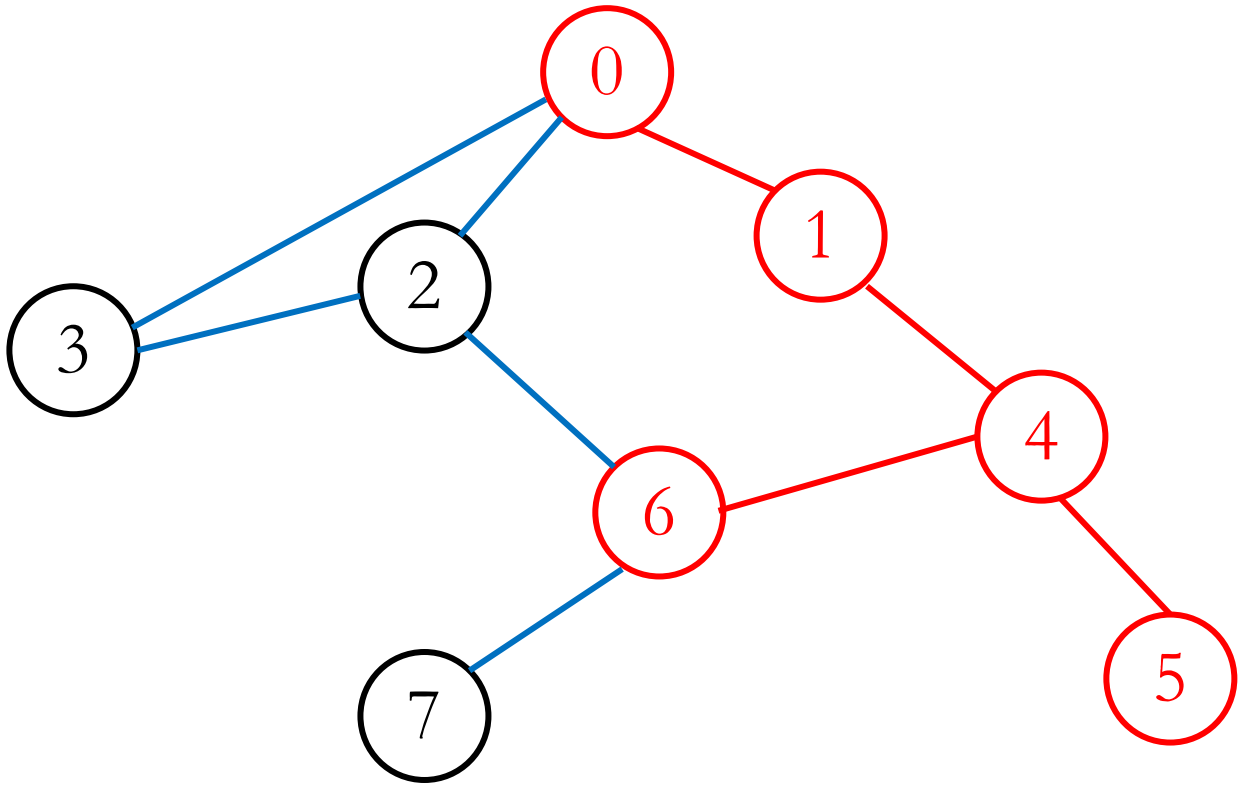
深さ優先探索（イメージ）

一歩戻って

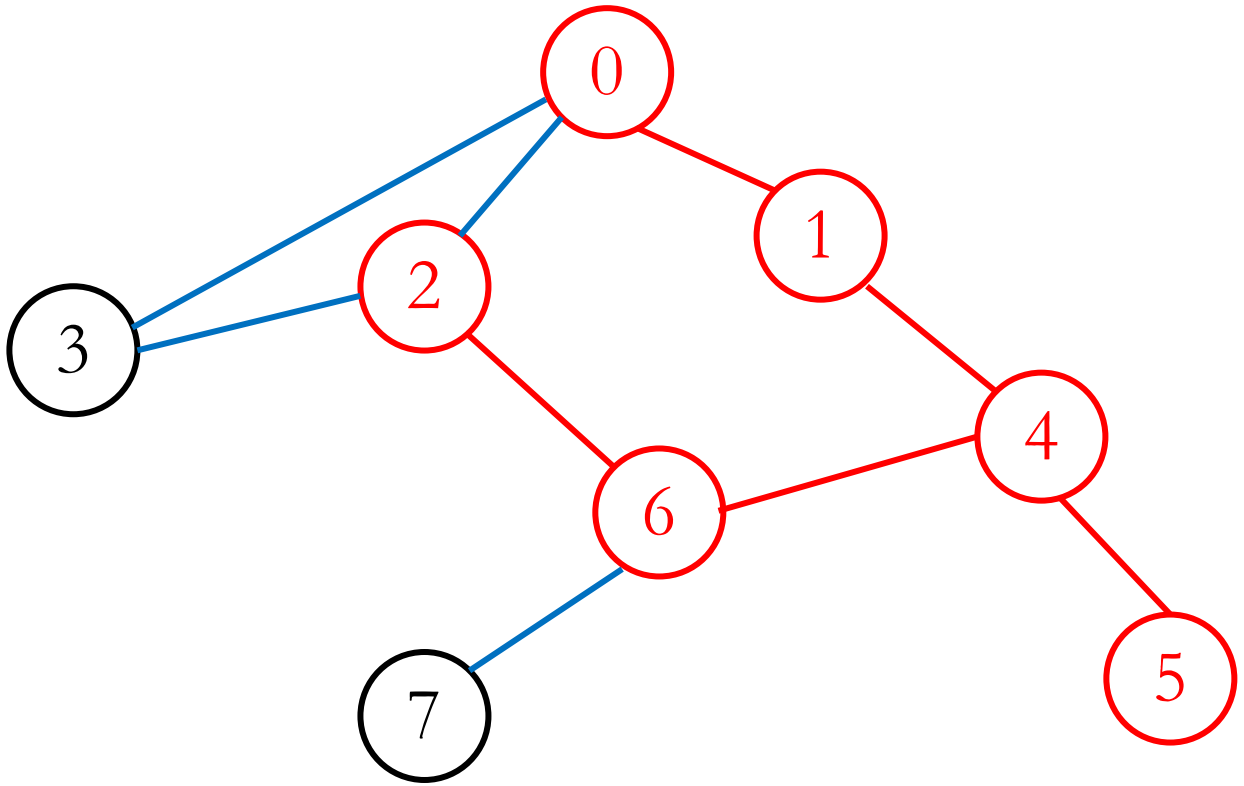


深さ優先探索（イメージ）

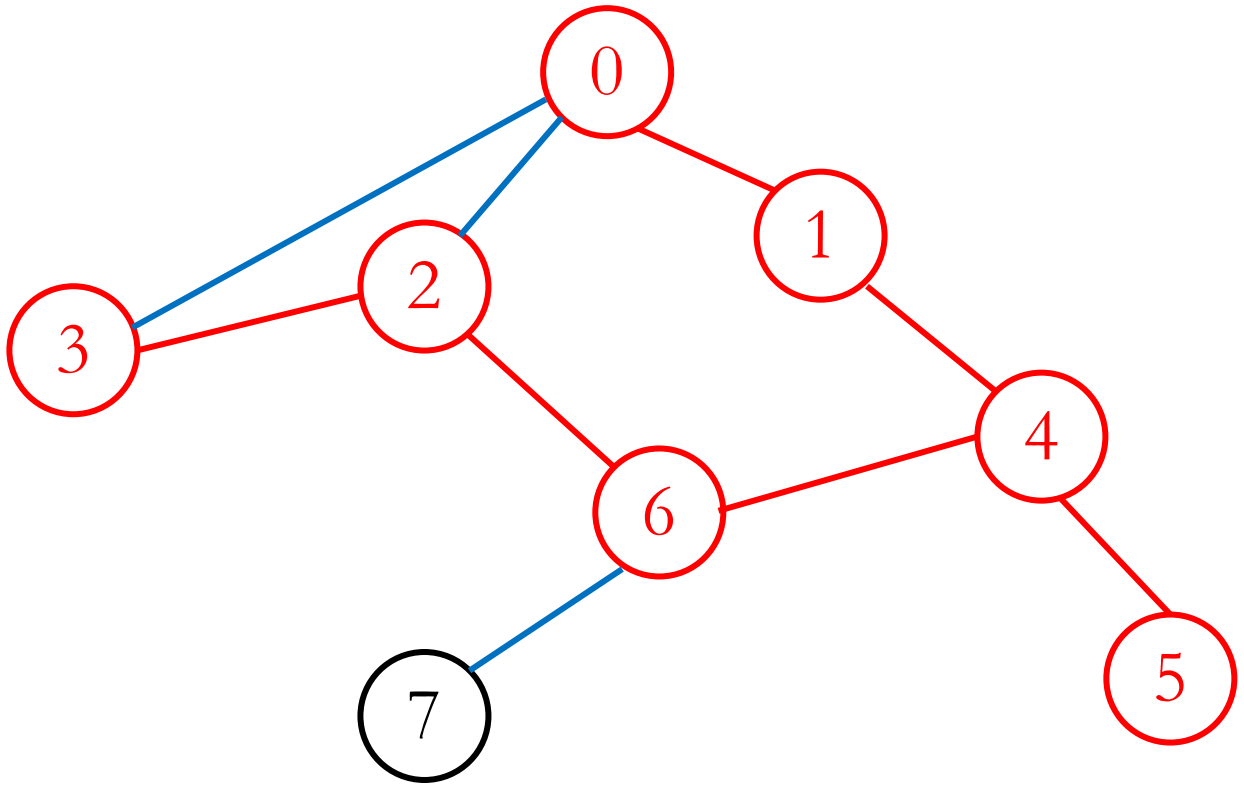
新しい道を探す



深さ優先探索（イメージ）

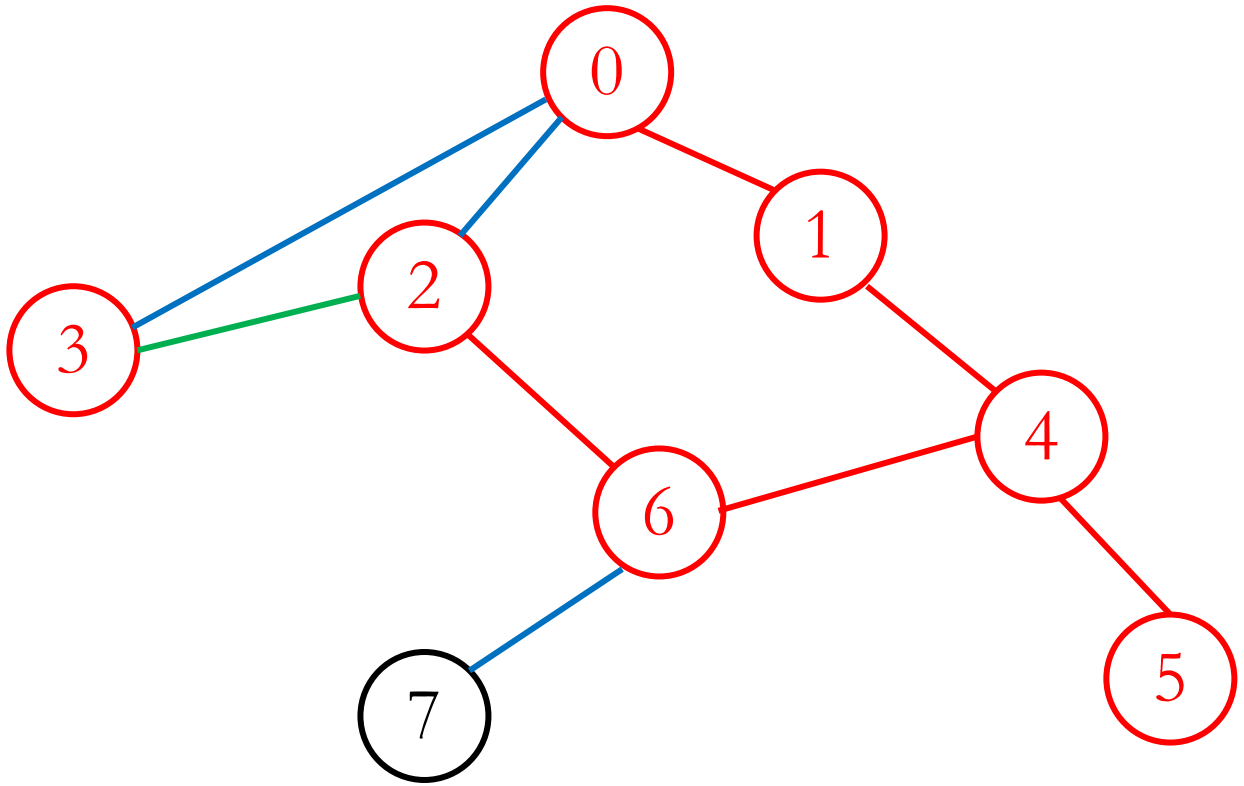


深さ優先探索（イメージ）



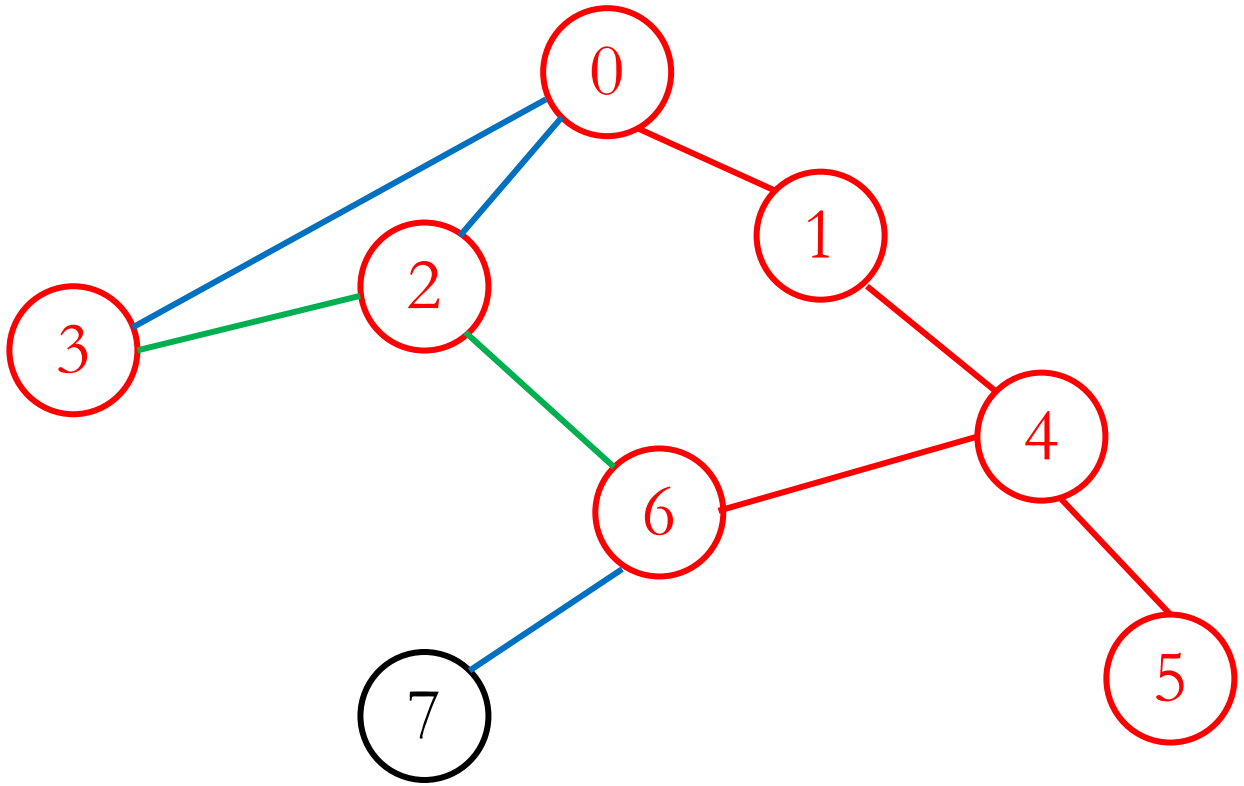
深さ優先探索（イメージ）

一歩戻って



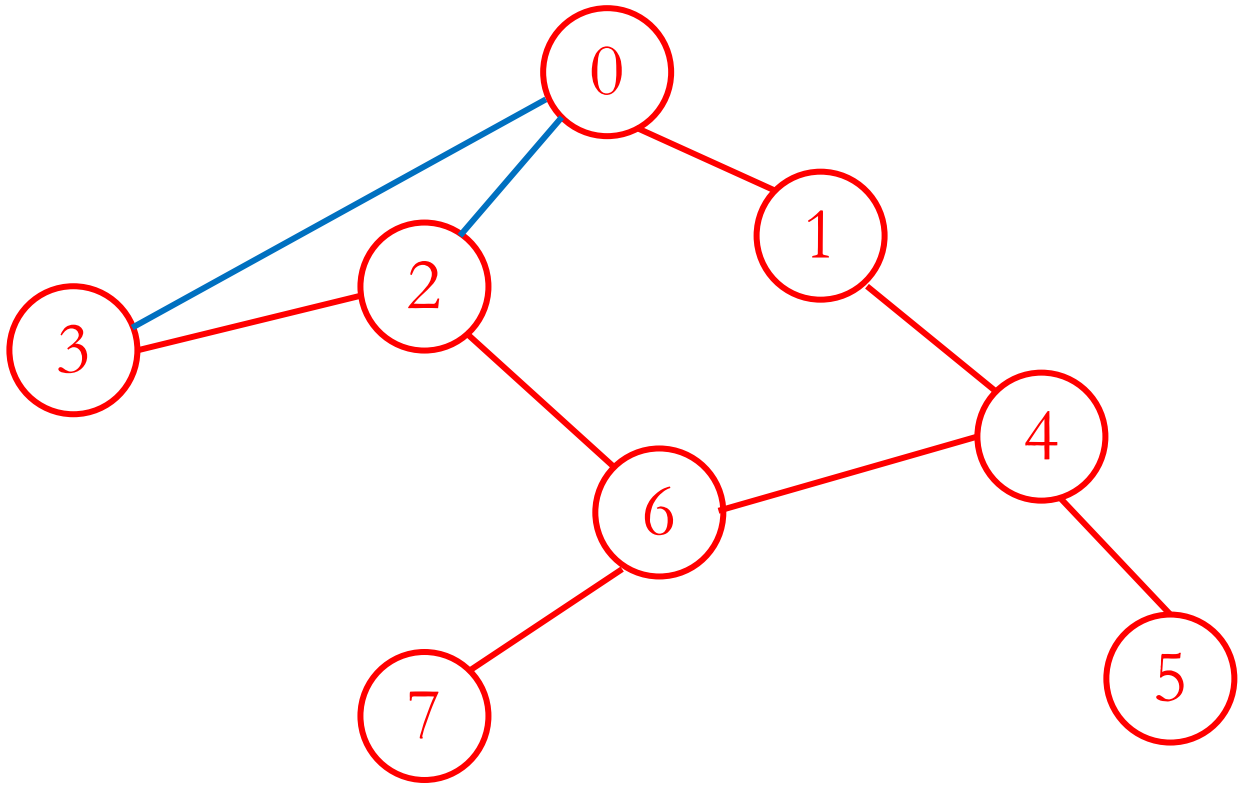
深さ優先探索（イメージ）

さらに一歩戻って

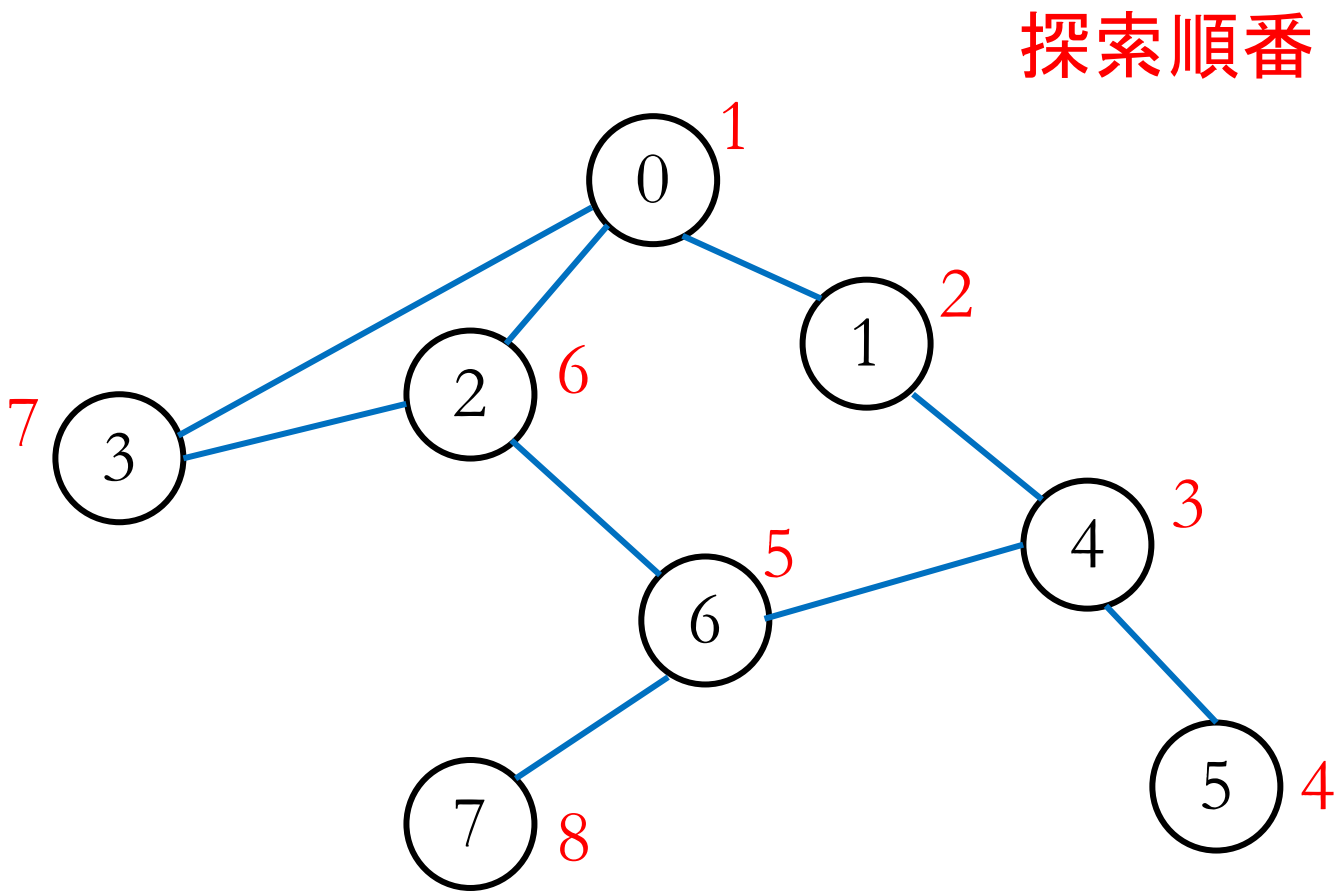


深さ優先探索（イメージ）

新しい道を探す



深さ優先探索（イメージ）



まとめ

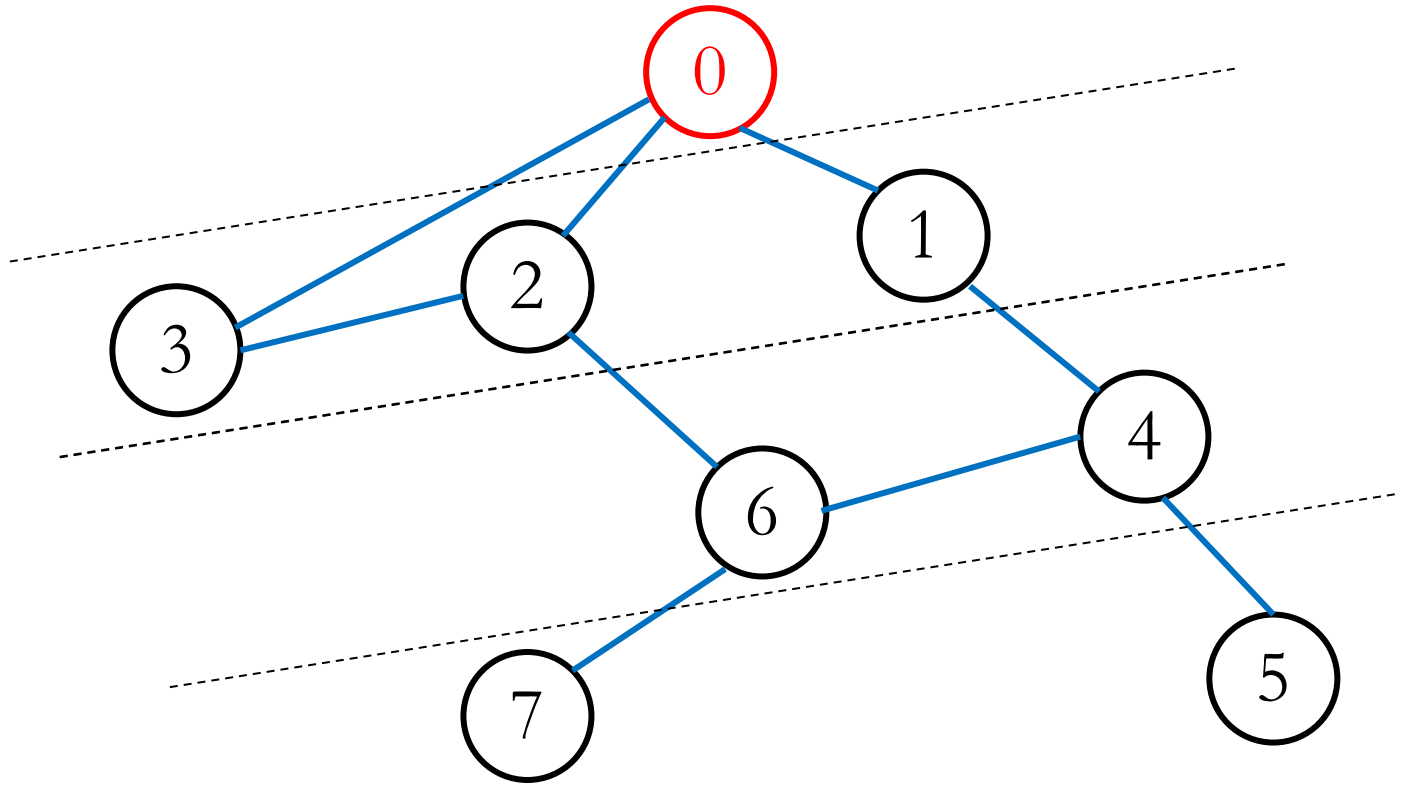
- 頂点0を始点とした深さ優先探索のノードの訪問（探索）順番は以下の通りである

  0 1 4 5 6 2 3 7

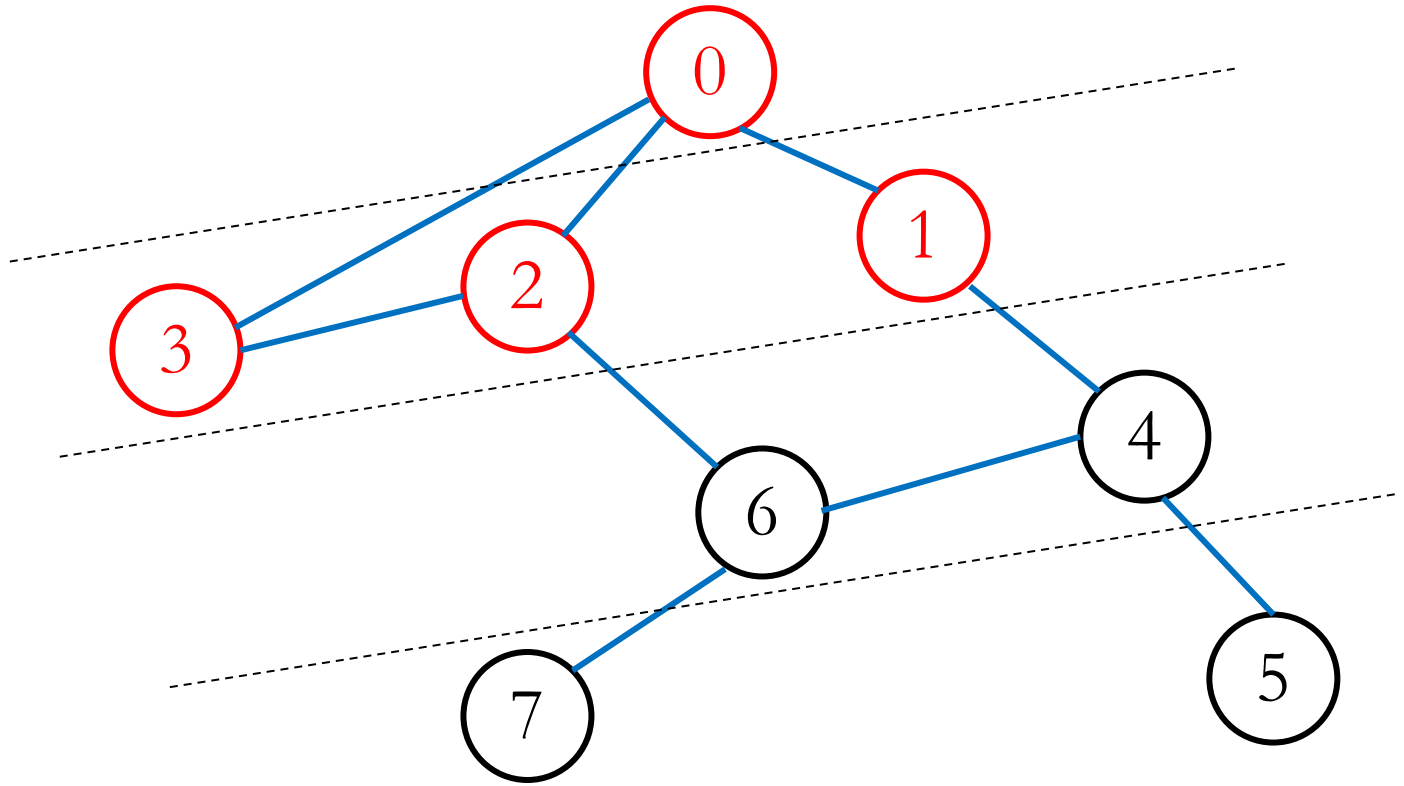
manaba小テスト：03-2

- 10分
- 8点

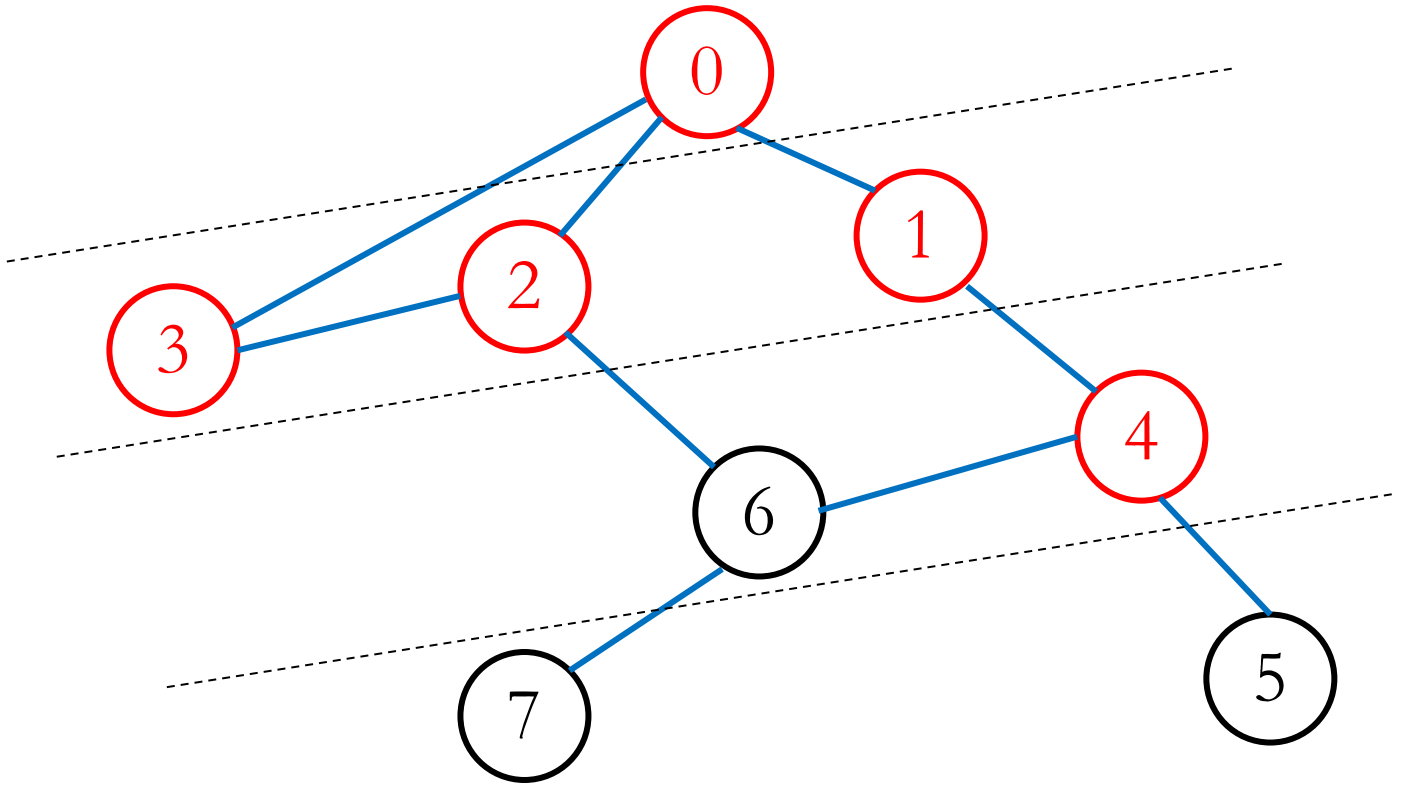
幅優先探索（イメージ）



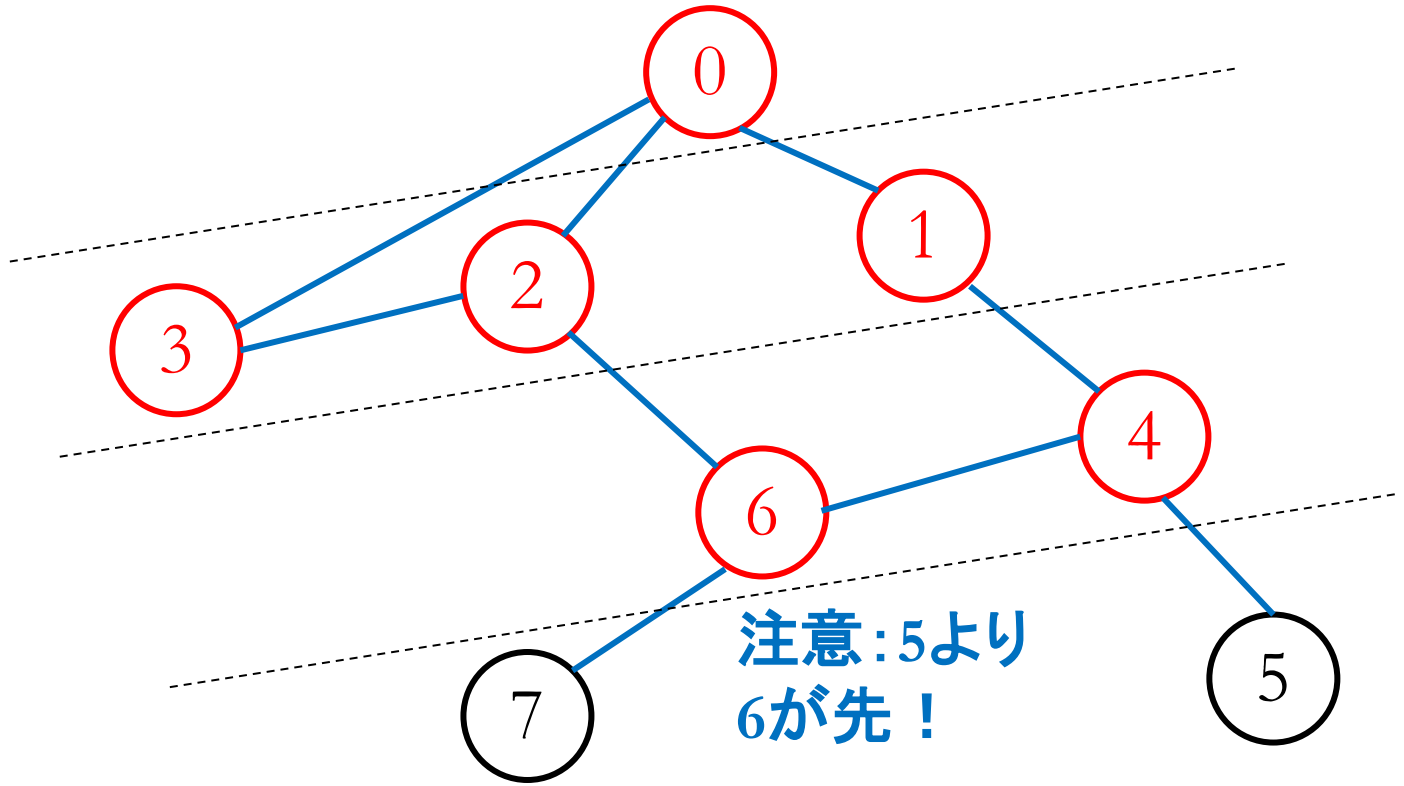
幅優先探索（イメージ）



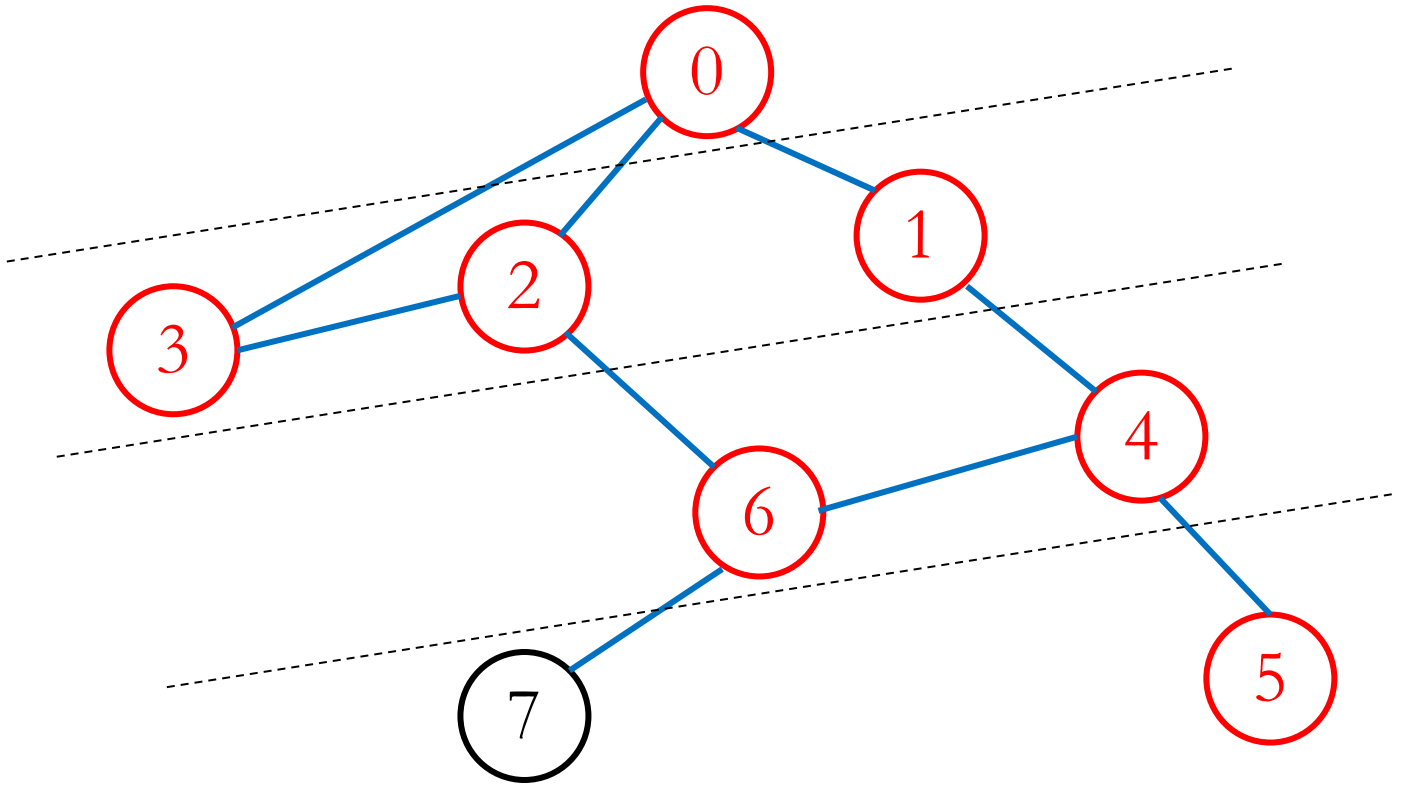
幅優先探索（イメージ）



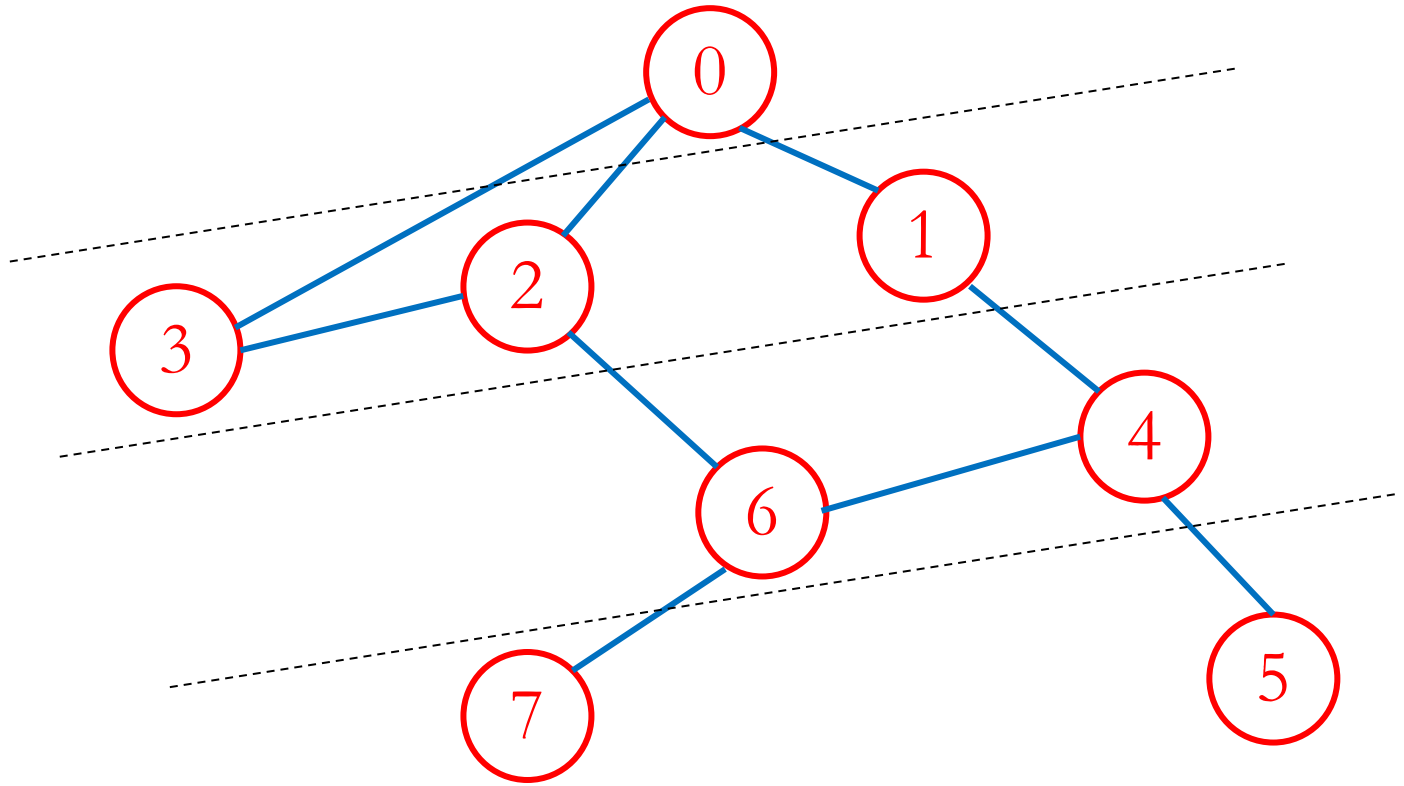
幅優先探索（イメージ）



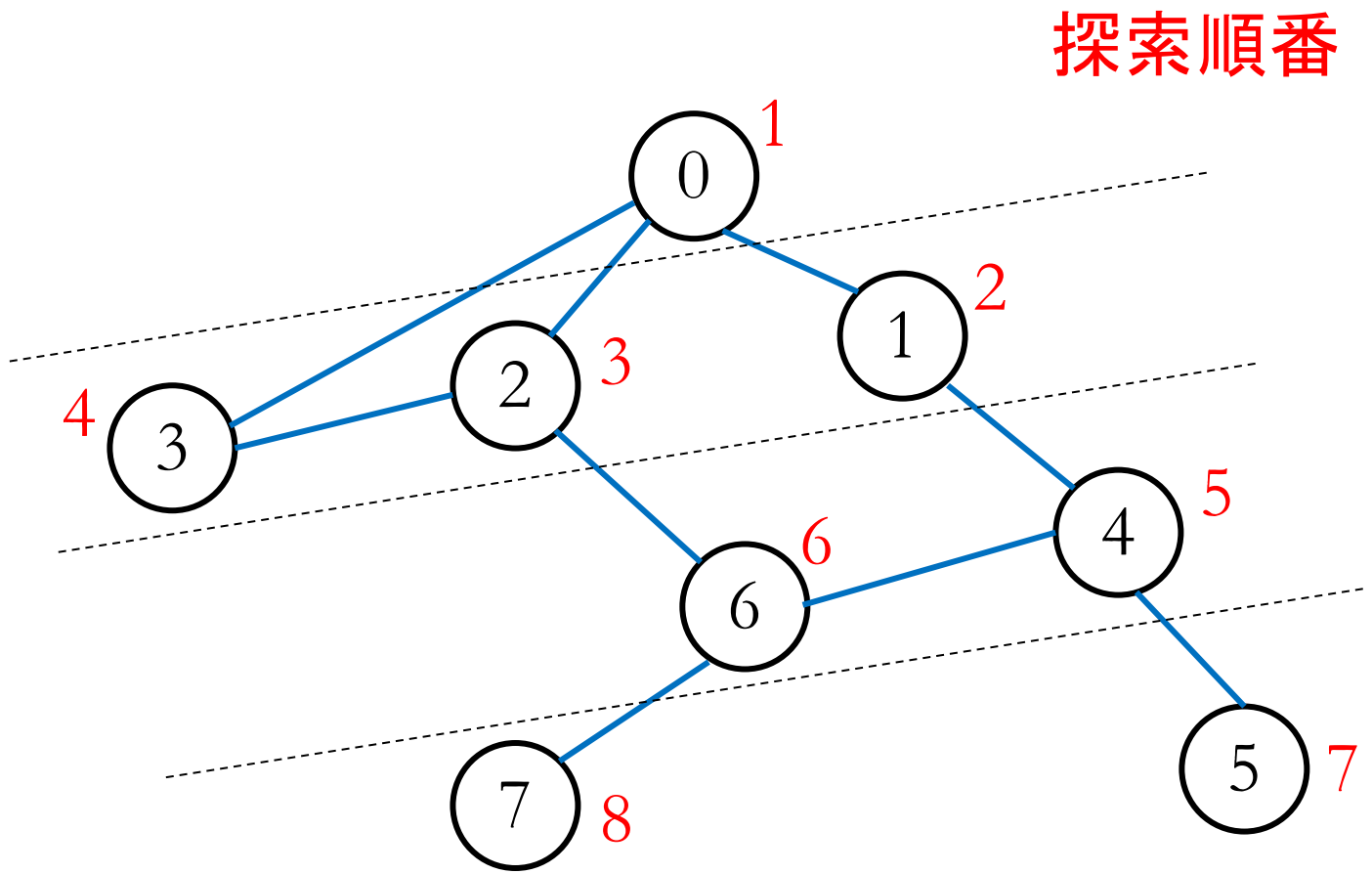
幅優先探索（イメージ）



幅優先探索（イメージ）



幅優先探索（イメージ）



manaba小テスト：03-3

- 8分
- 6点