アルゴリズムとデータ構造

第12回
グラフの深さ優先探索

塩浦昭義

情報科学研究科 准教授

shioura@dais.is.tohoku.ac.jp

http://www.dais.is.tohoku.ac.jp/~shioura/teaching

期末試験について

- 日時：2月8日（水）13:00〜14:30（確定）
- 受験資格：
  - 中間試験に合格
  - 中間試験以降にレポートを一回以上提出
- 教科書，ノート等の持ち込みは一切不可
- 座席はこちらで指定
- 試験内容：第7回（アルゴリズムの設計）〜第13回（最終回）の講義で教えたところ
- 50点満点，29点以下は追試レポートもしくは単位不可

※ 来週2月1日（水）は講義（最終回）を行います

※ 期末試験の出来が悪く，かつ救済を希望する場合は 2月10日（金）12:00までに問い合わせること （救済できない可能性もあり）

無向グラフの2連結成分

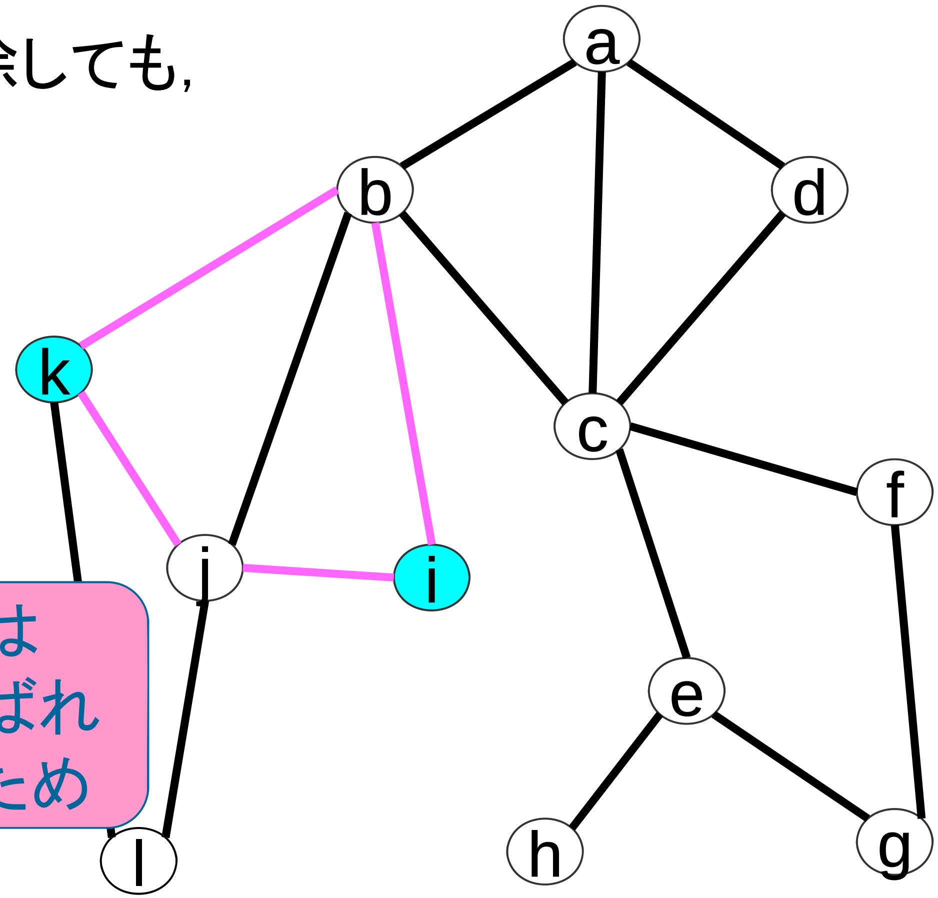
- 無向グラフG＝(V, E)において,
  頂点 u, v は同じ2連結成分に含まれる
  ⇔u, v 以外の頂点 w を削除しても,
  uからvへの路が存在

k と i は同じ
2連結成分に含まれる

a と c は同じ
2連結成分に含まれる

e と h は同じ
2連結成分に含まれる

eとhは
枝で結ばれ
ているため



無向グラフの2連結成分

- 無向グラフG＝(V, E)において,

頂点 u, v は同じ2連結成分に含まれる

←→u, v 以外の頂点 w を削除しても,

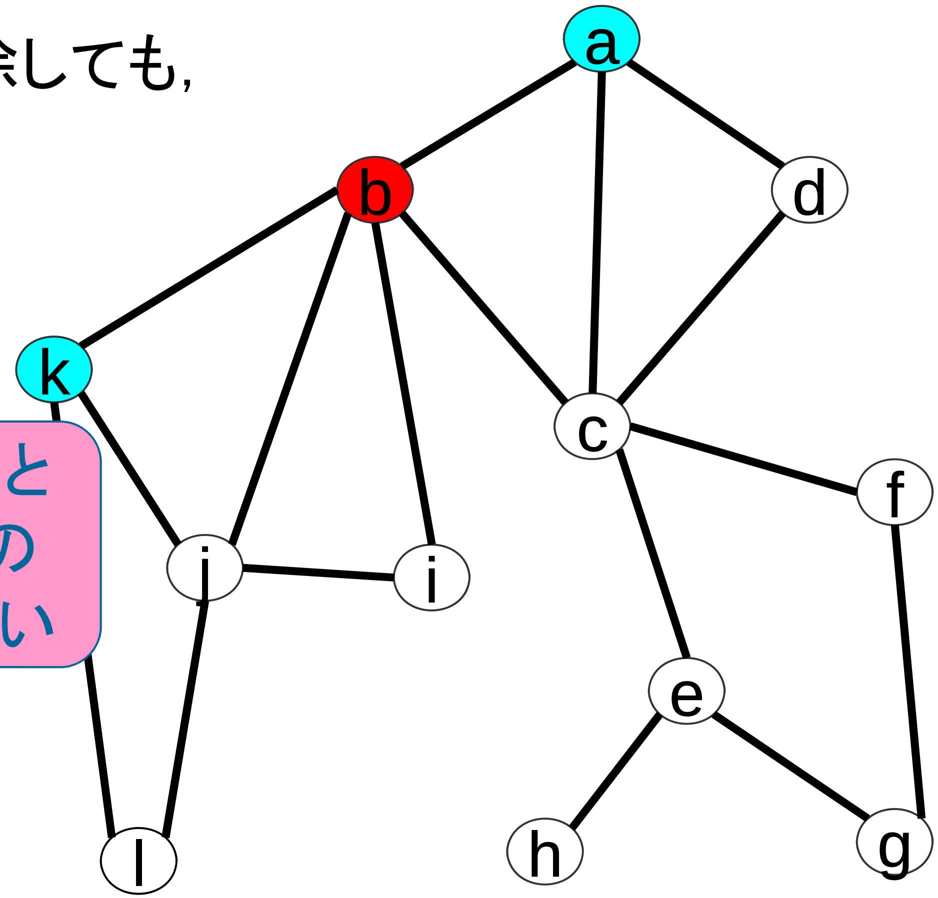
uからvへの路が存在

a と k は同じ
2連結成分に含まれない

c を削除すると
b から g への
路が存在しない

b を削除すると
a から k への
路が存在しない

b と g は同じ
2連結成分に含まれない

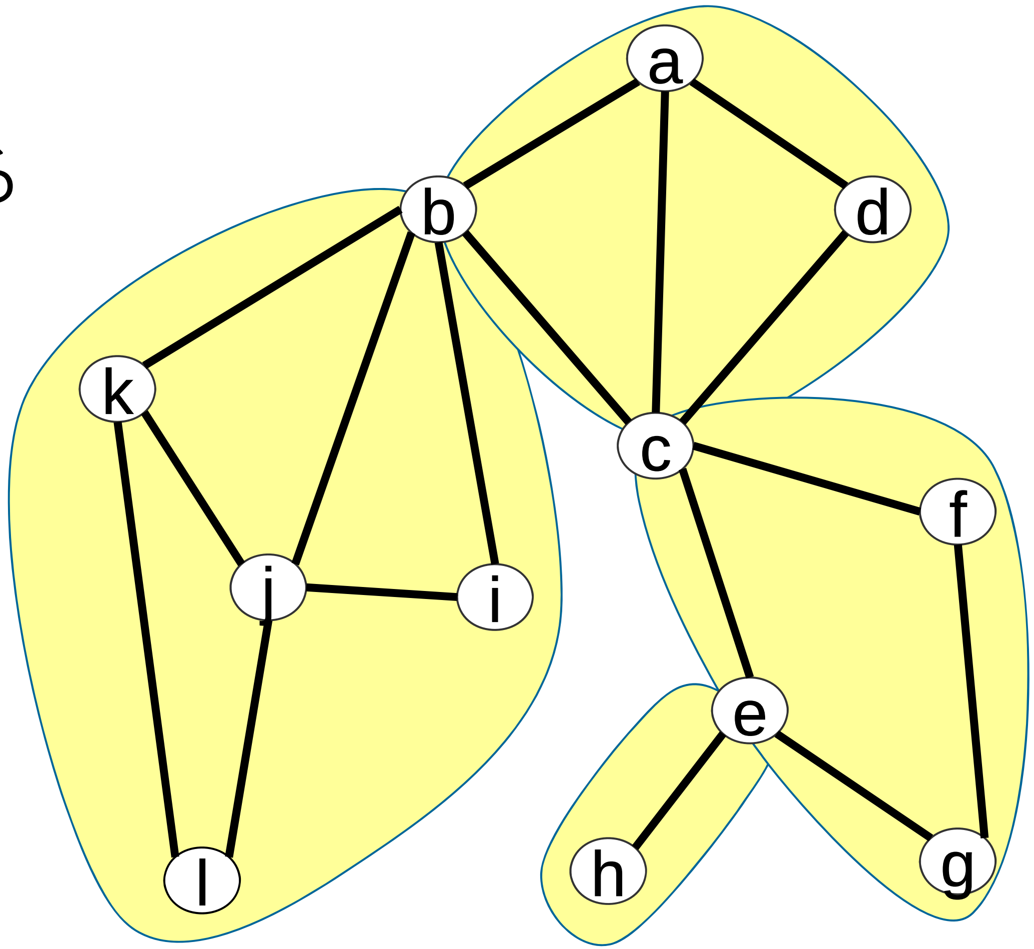


２連結成分分解と関節点

- 同じ連結成分に含まれる頂点をグループ分け
  ➔２連結成分分解

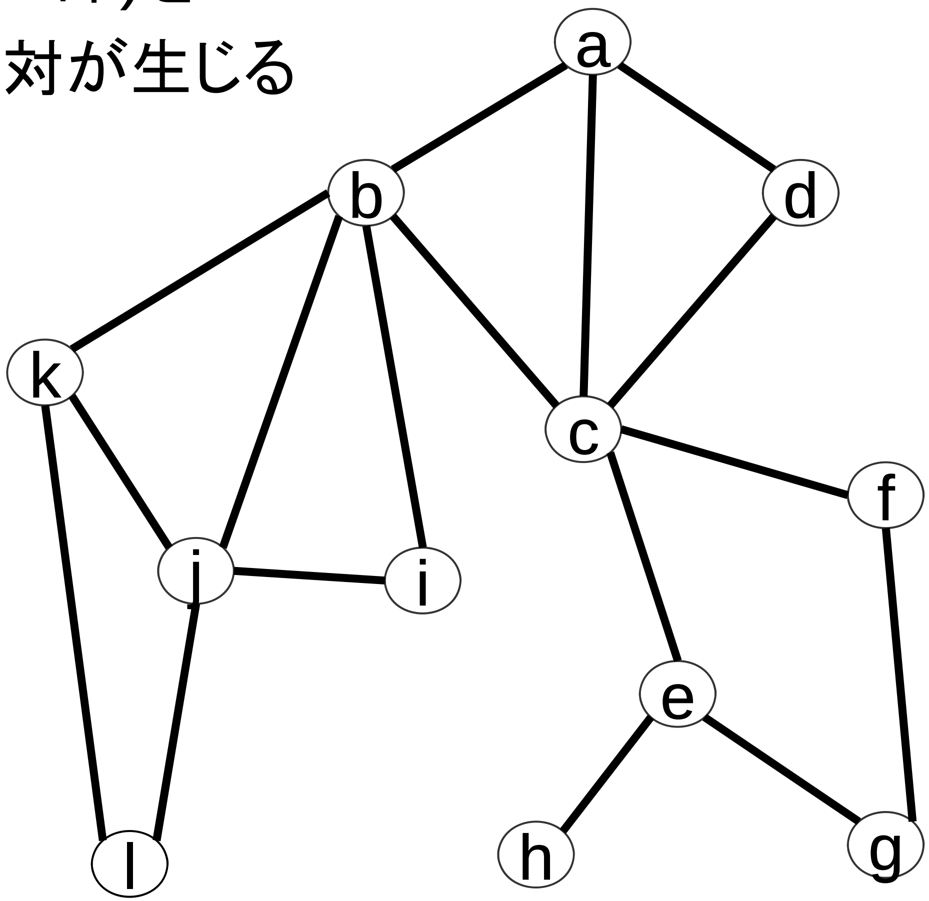
- 複数の２連結成分に含まれる
  頂点が存在➔関節点と呼ぶ

頂点 b, c, e は関節点



無向グラフの関節点

- **性質**: 無向グラフG＝(V, E)において，頂点 u は関節点
  ←→u (および u に接続する枝全部)を
  削除すると，非連結になる頂点対が生じる



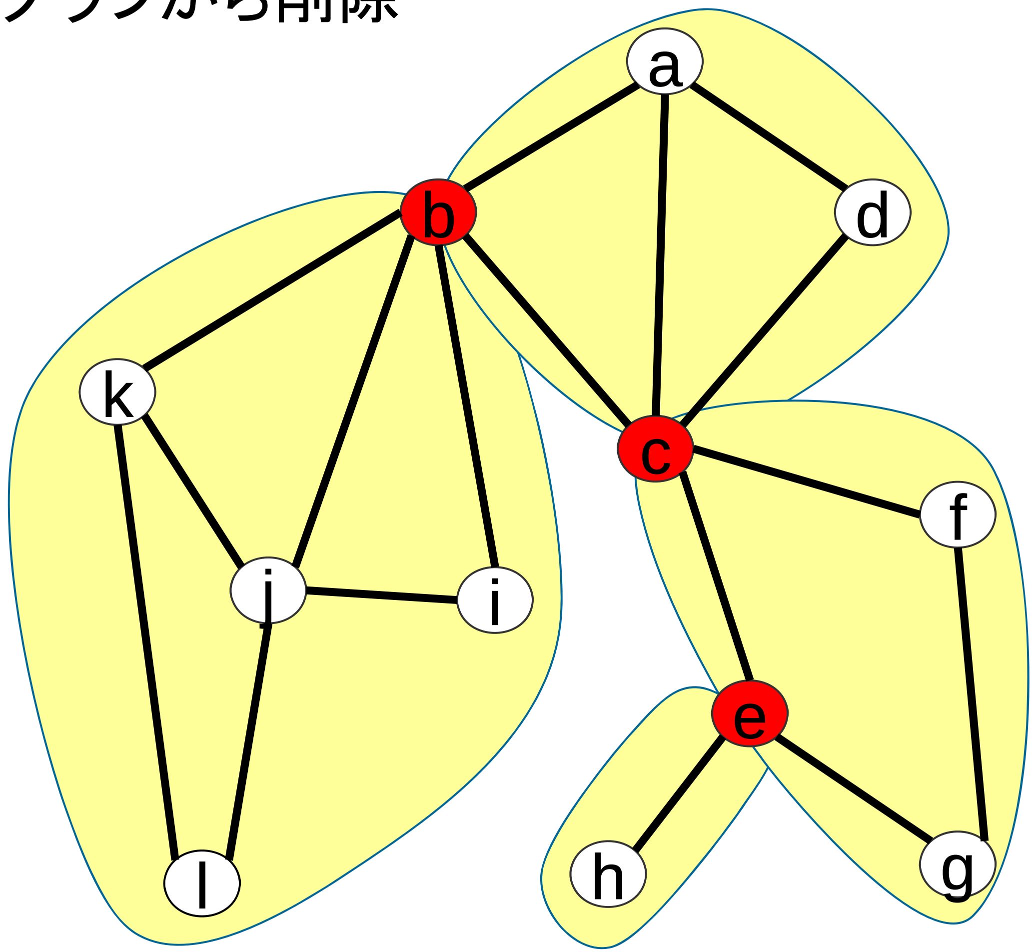
頂点 b, c, e は関節点
他の頂点は関節点ではない

関節点から2連結成分分解を求める

関節点が計算できれば，２連結成分分解も計算できる

（１）関節点および接続する枝をグラフから削除

（２）連結成分に分解



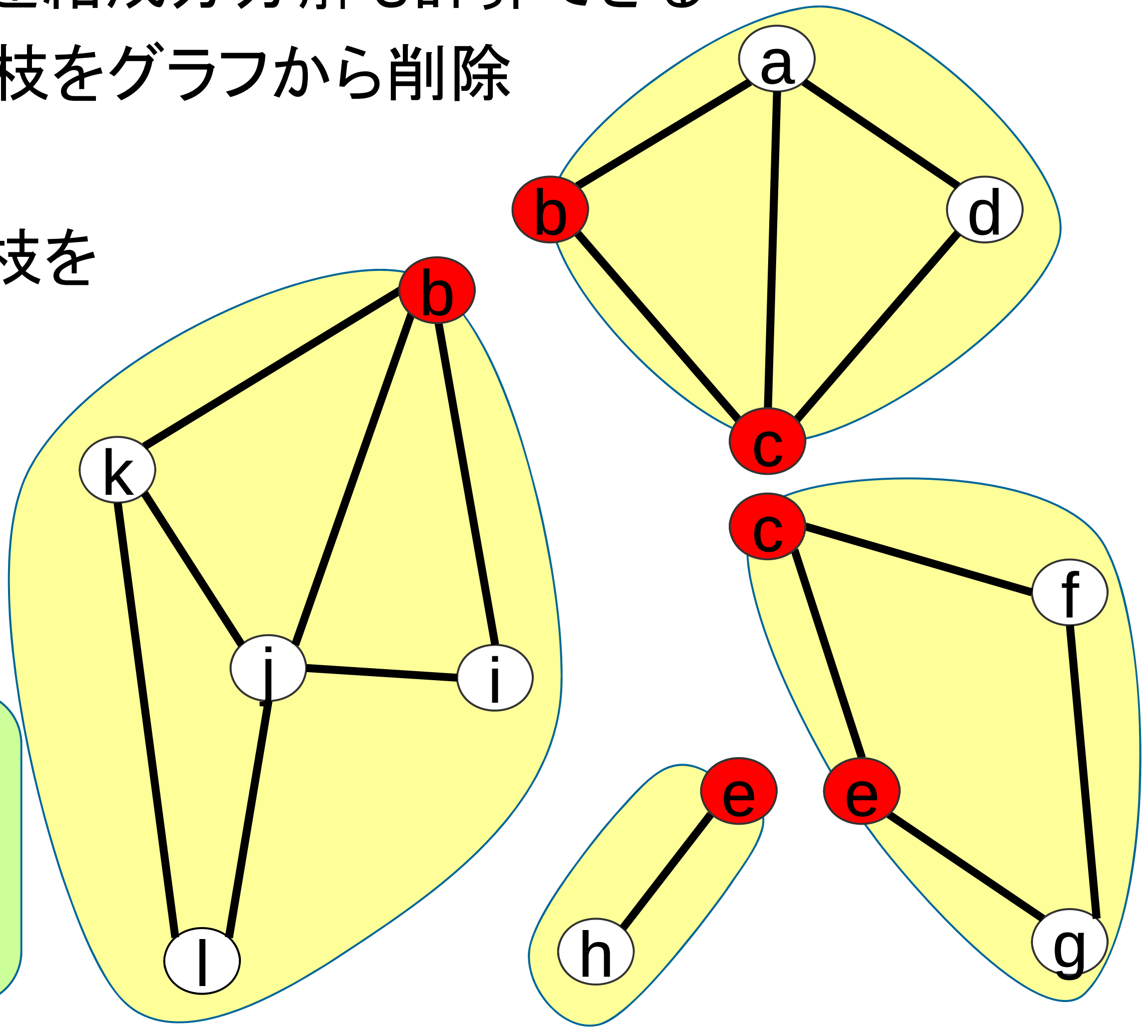
関節点から2連結成分分解を求める

関節点が計算できれば，２連結成分分解も計算できる

（１）関節点および接続する枝をグラフから削除

（２）連結成分に分解

（３）関節点に接続していた枝を
　　各連結成分に戻す



関節点が与えられれば，
計算時間は O(m＋n)

深さ優先探索を使って
全ての関節点を
O(m＋n)時間で求める
方法を説明する

関節点と lowpt

num[u]：頂点 u が何番目に走査されたかを表す数字

lowpt[u]

＝ min{num[v] | v=u, もしくは

頂点uの子孫wに対し,

Tに含まれない枝(v,w)が存在}

例1： 頂点 j の

子孫 k に枝(b,k)が接続

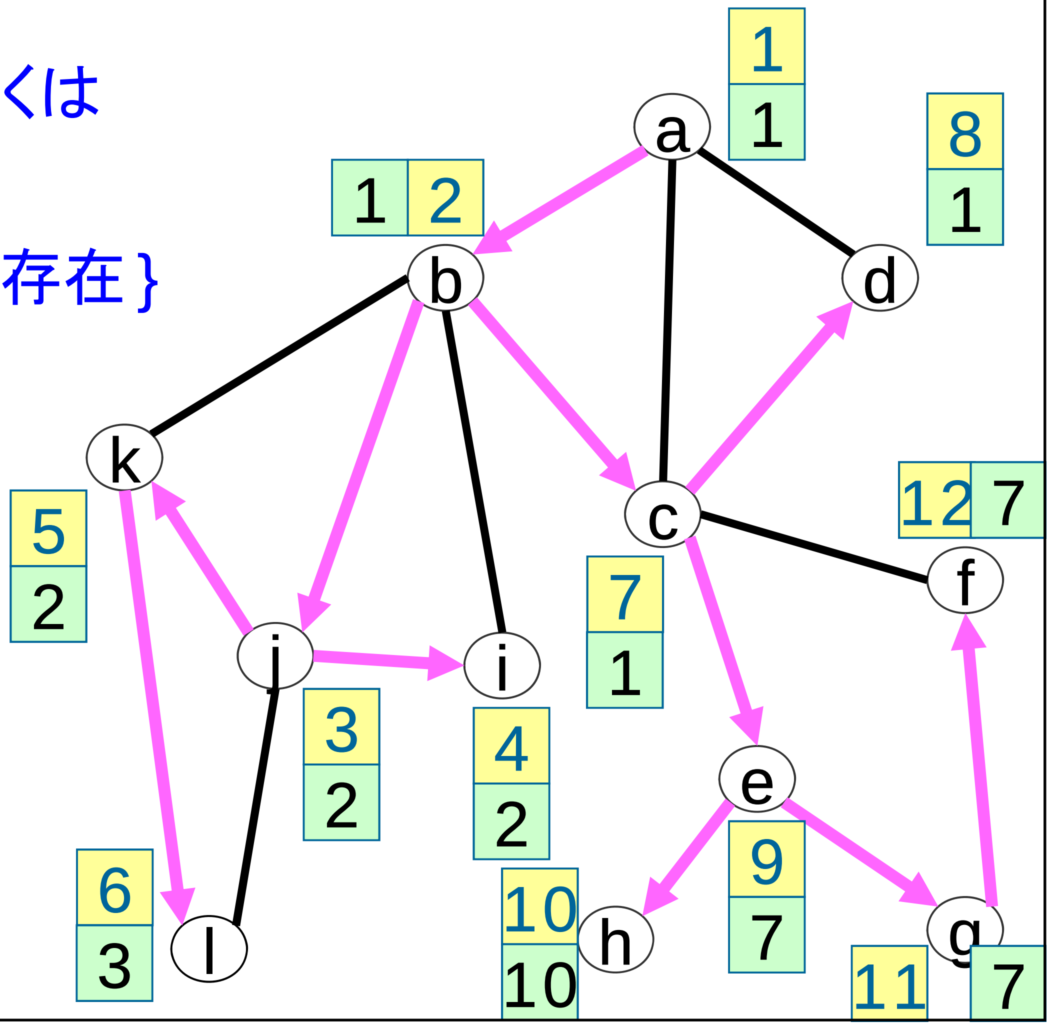
子孫 i に枝(b,i)が接続

➔ lowpt[j]=num[b]=2

例2： 頂点 e の

子孫 f に枝(c,f)が接続

➔ lowpt[e]=num[c]=7

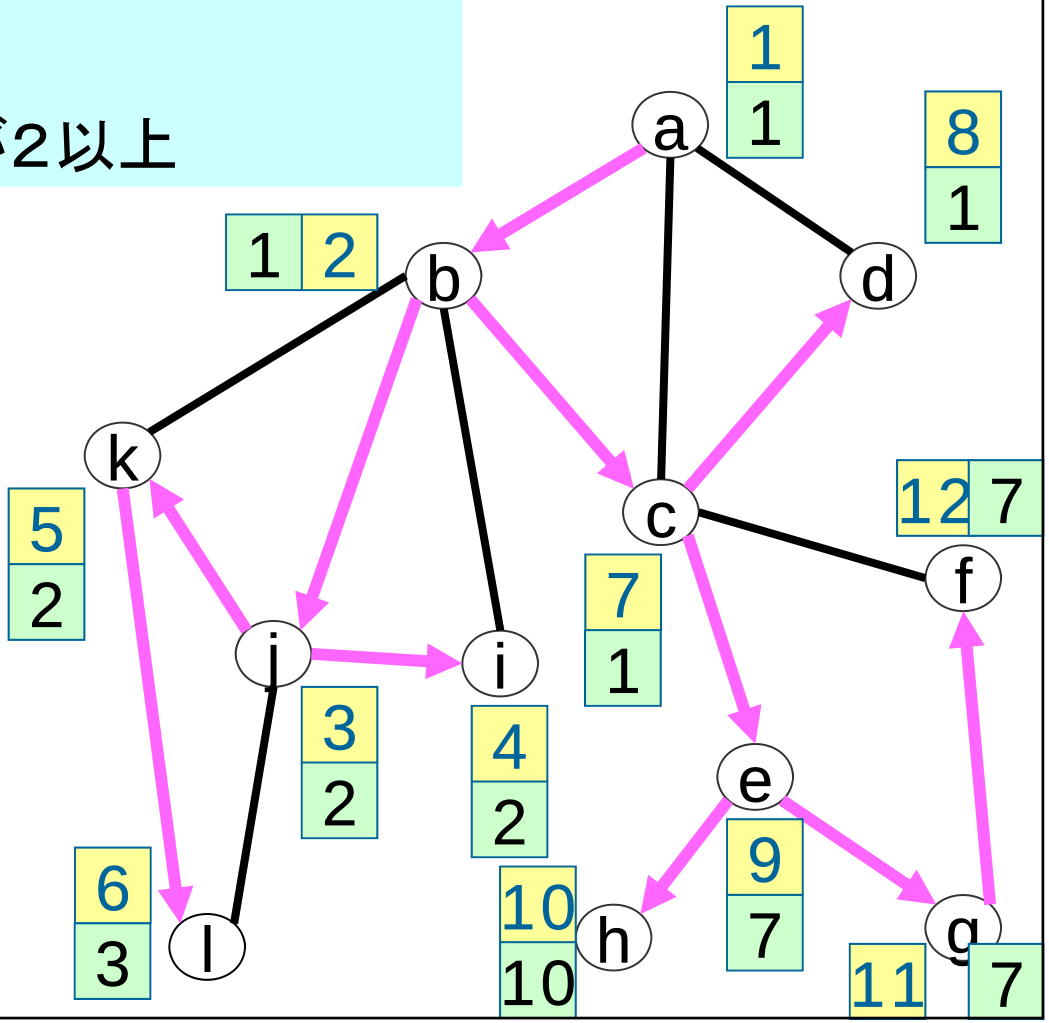


関節点と lowpt に関する性質

定理: u は関節点
←→u の子供 v が存在して次の条件を満たす
(i) lowpt[v]≧num[u]
(ii) u が根のとき，子供の数が2以上

例1: 頂点 b と子供 j に対し
(i)lowpt[j]=2=num[b]
(ii) 頂点bは根ではない
→ b は関節点

例2: 頂点 c と子供 eに対し
(i)lowpt[e]=7=num[c]
(ii) 頂点cは根ではない
→ c は関節点

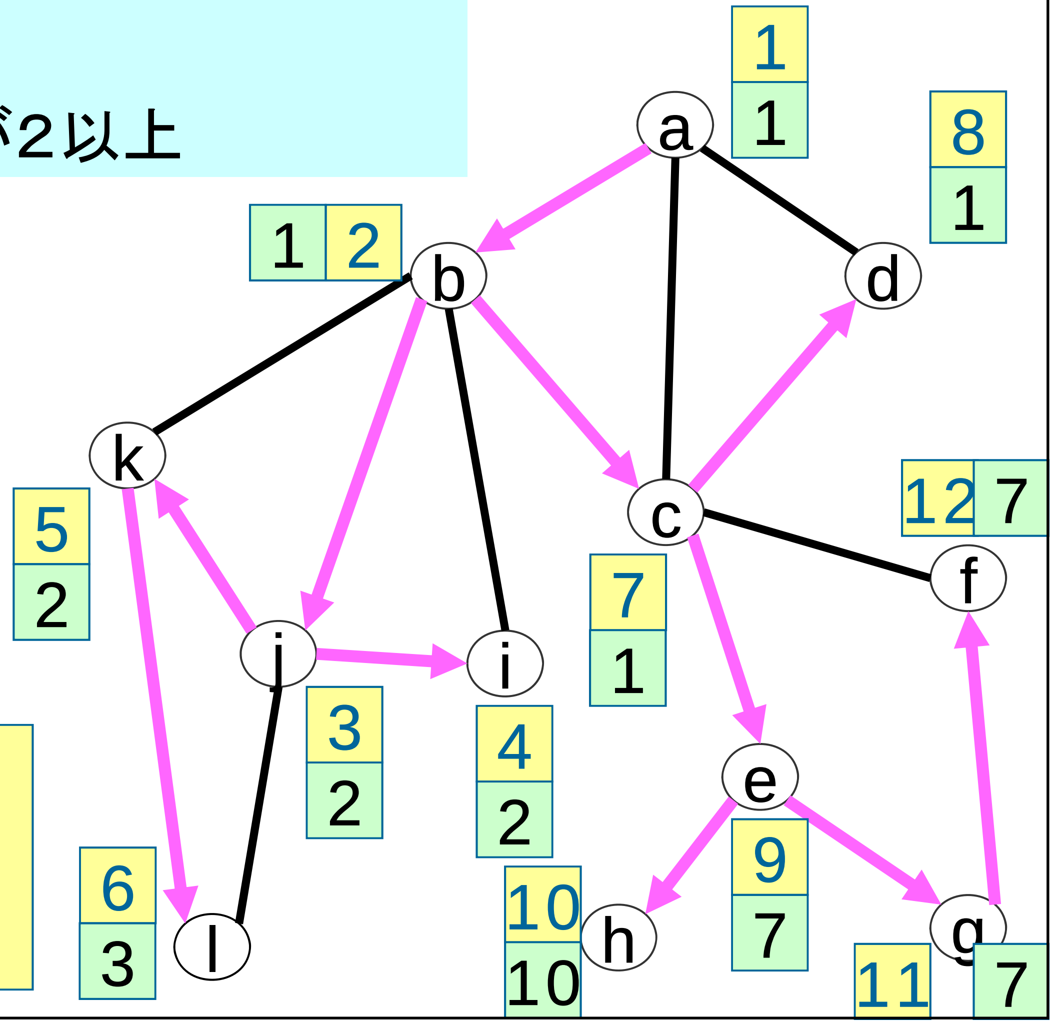


関節点と lowpt に関する性質

定理: u は関節点
←→u の子供 v が存在して次の条件を満たす
(i) lowpt[v]≧num[u]
(ii) u が根のとき，子供の数が2以上

例3: 頂点 j の子供はk, i
lowpt[k]=2<3=num[j]
lowpt[i]=2<3=num[j]
➔ j は関節点ではない

例4: 頂点 a の子供は b
(ii) b以外の子供は存在しない
➔ a は関節点ではない

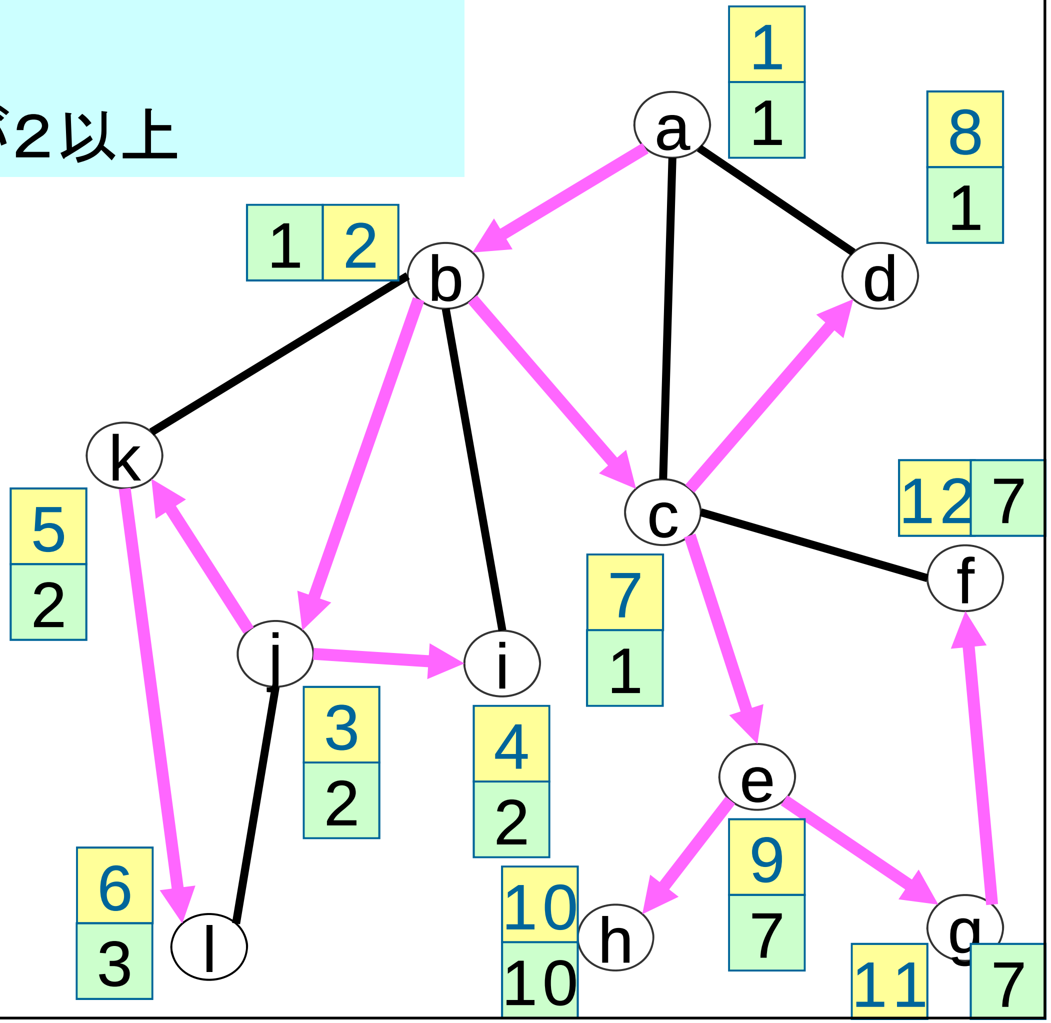


関節点の計算

定理: u は関節点
←→u の子供 v が存在して次の条件を満たす
(i) lowpt[v]≧num[u]
(ii) u が根のとき，子供の数が2以上

深さ優先探索を使うことによって
- lowpt[v]全てをO(m+n)時間で計算できる
- 根である頂点の子供の数をO(m+n)時間で計算できる

➔定理を使うことにより，関節点をO(m+n)時間で計算可能



有向グラフの深さ優先探索

無向グラフの場合と基本的に同じ

- 各頂点，各枝を白く塗る
- 各頂点 u∈V に対し，
  u が白色（未走査）ならば
  手続きDFS-VISIT(u)を実行

手続き DFS-VISIT(u)

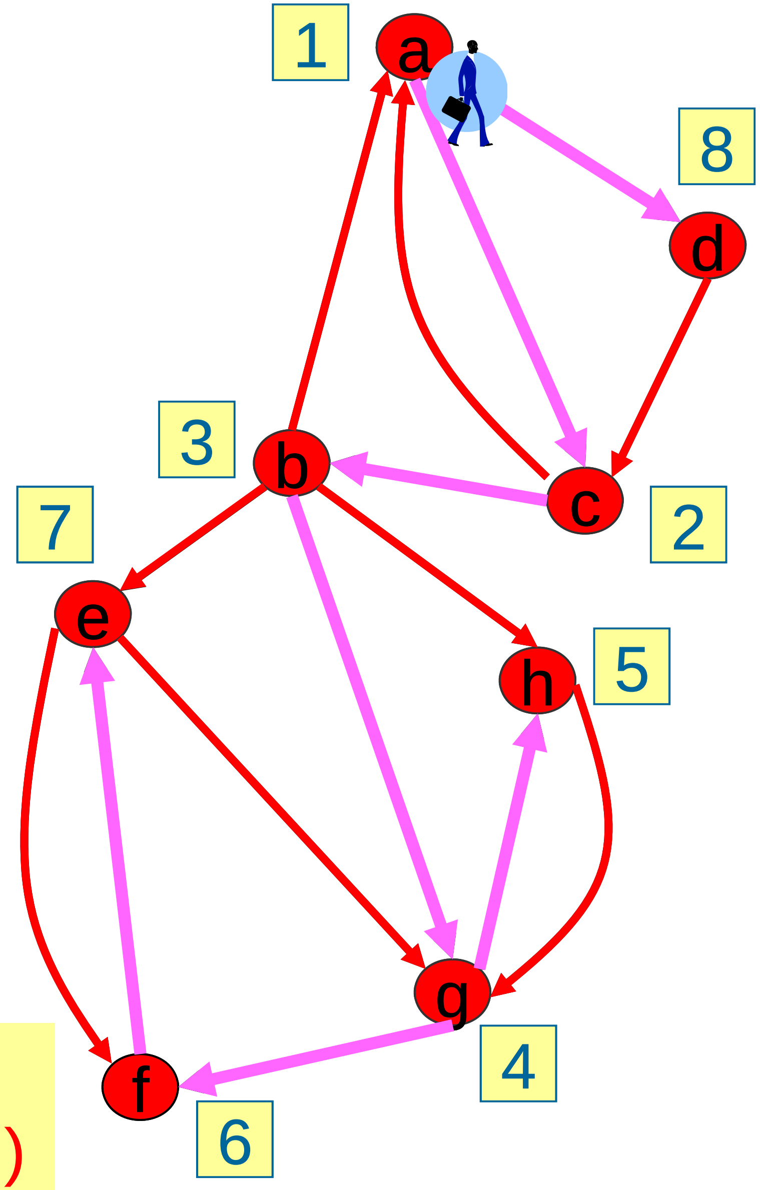
(a) u を黒く塗る

(b) u に接続する各枝 (u, v) に対し，以下を実行：

枝が白色（未走査）ならば，黒く塗る

v が白色（未走査）ならば

DFS-VISIT(v) を再帰呼び出し



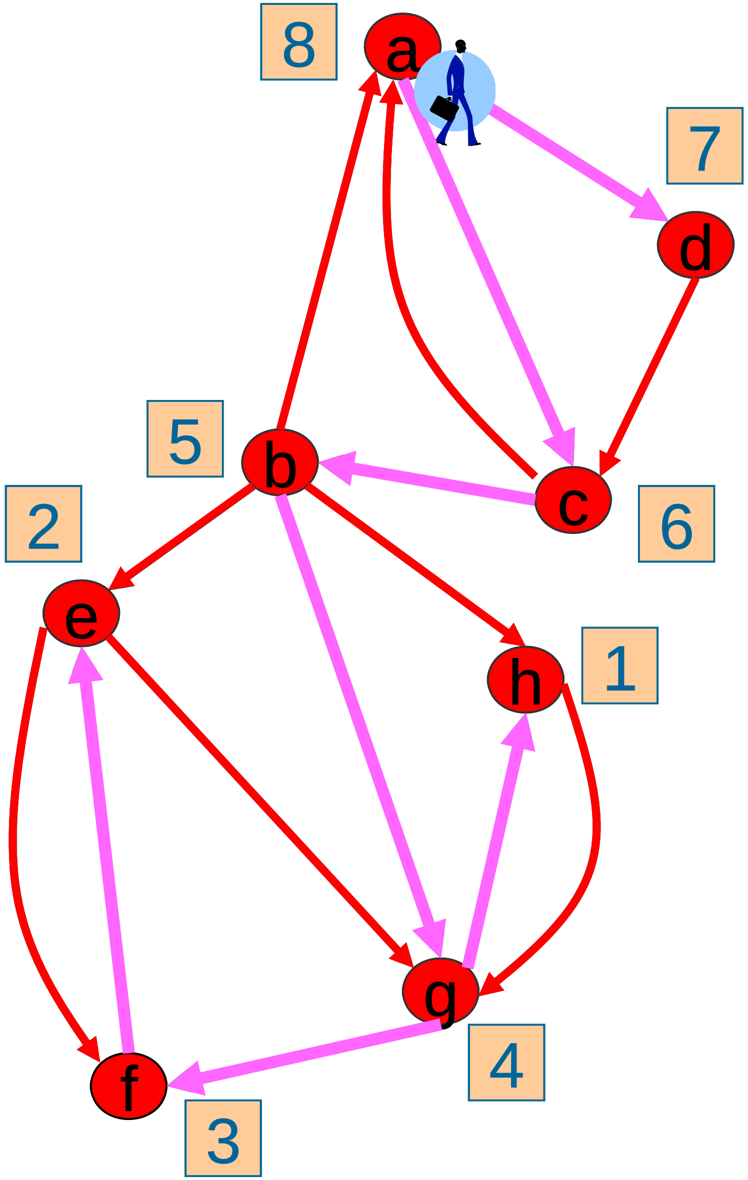
深さ優先探索の実行時間：
グラフを隣接リストで表現するとO(m＋n)

有向グラフの深さ優先探索

頂点を初めて訪問した時間の順に,
各頂点に番号を付ける
➔ いろいろと便利

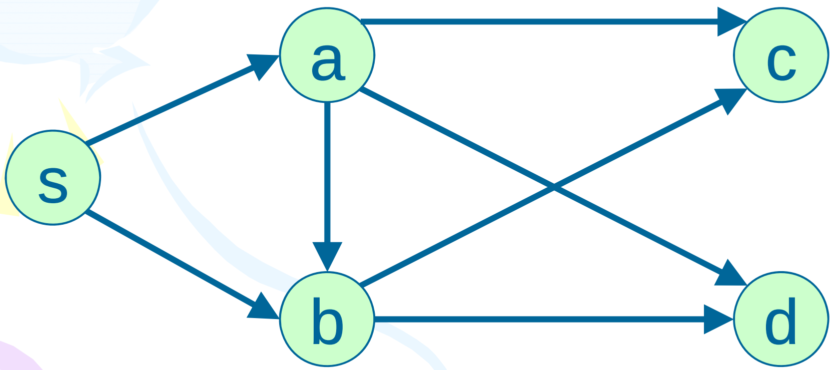
同様に,

頂点を最後に訪問した時間の順に,
(頂点から出る枝を全て調べ終えた順に)
各頂点に番号を付ける
➔ いろいろと便利

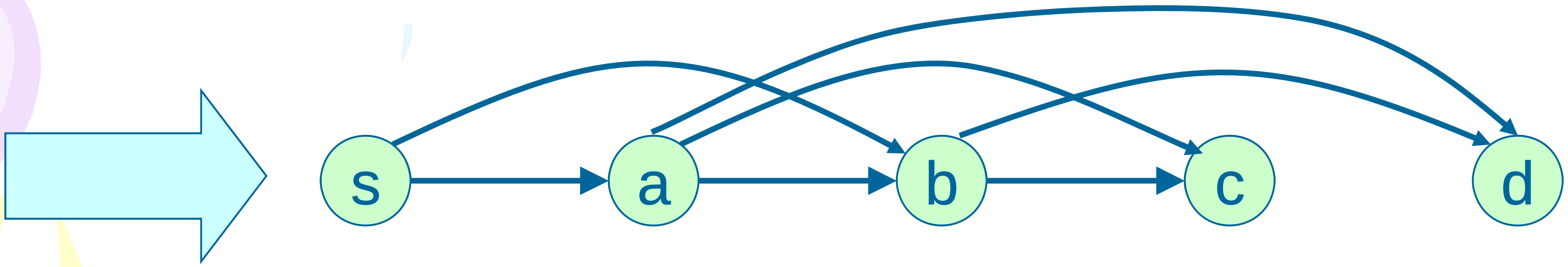


閉路のない有向グラフのトポロジカルソート

- 閉路のない有向グラフに対し，以下の条件を満たすように頂点を並べることが可能
  - 各枝は，必ず左から右に向かう
- このように頂点を並べること➔トポロジカルソート
- 応用：複数の作業のスケジューリング



閉路のない有向グラフ

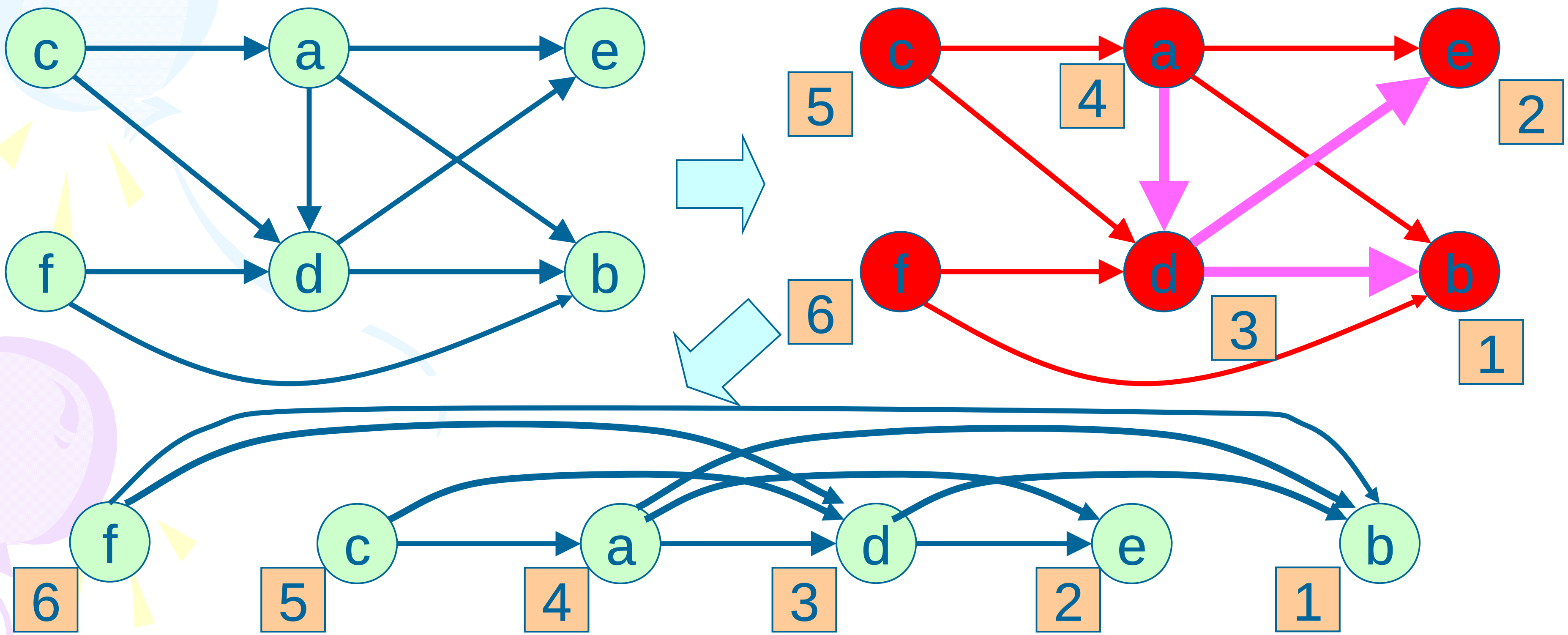


トポロジカルソートを求めるアルゴリズム

深さ優先探索を利用➔トポロジカルソートが計算できる

(1) グラフに深さ優先探索を適用し，
　　最後に訪問した時間により頂点を番号付けする

(2) 番号の大きい方から小さい方に並べる



トポロジカルソートを求めるアルゴリズムの正当性

グラフに深さ優先探索を適用し，最後に訪問した時間により頂点を番号付けする　➔各頂点vの番号を f(v)と書く

全ての枝 (u,v) に対して f(u) ＞ f(v) ならばOK

ヨ (u,v)： f(u) ＜ f(v) と仮定，矛盾を導く

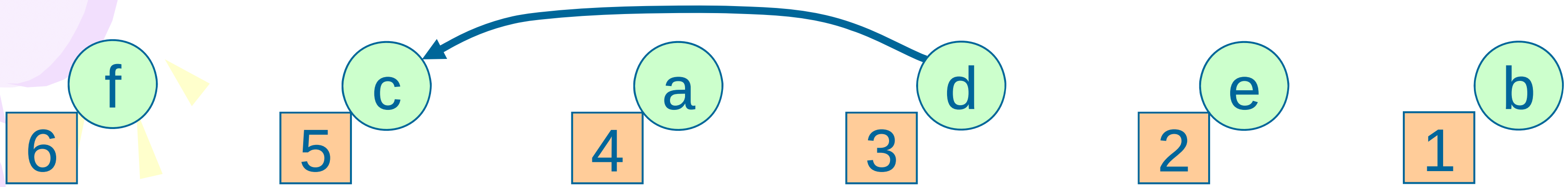
枝(u,v)を探索したとき，

(a) v は未訪問➔ vはuの子孫➔f(u)>f(v) が成り立つ（矛盾）

(b) v は訪問終了後

　➔ uの訪問は終了していないので f(u)>f(v) が成り立つ（矛盾）

(c) v は訪問中➔ v は u の先祖➔u, v を含む閉路が存在（矛盾）



強連結成分分解の計算

- 深さ優先探索を2回行う
- 1回目：普通に深さ優先探索を行い，各頂点uに対して最後に訪問した時間の順に番号を付ける←f(u)とする
- 2回目：
  - グラフの枝を全て逆向きにする
  - 右の深さ優先探索（修正版）を実行
  - 同じラベルごとに分解すると，強連結成分になっている

- 各頂点，各枝を白く塗る
- 各頂点 u∈V に対し，
  f(u)の大きい順に以下を実行
  u が白色（未走査）ならば
  k＝u とおき，
  手続きDFS-VISIT(u)を実行

手続き DFS-VISIT(u)

(a) u を黒く塗り，ラベルkを付ける

(b) u に接続する各枝 (u, v) に対し，以下を実行：

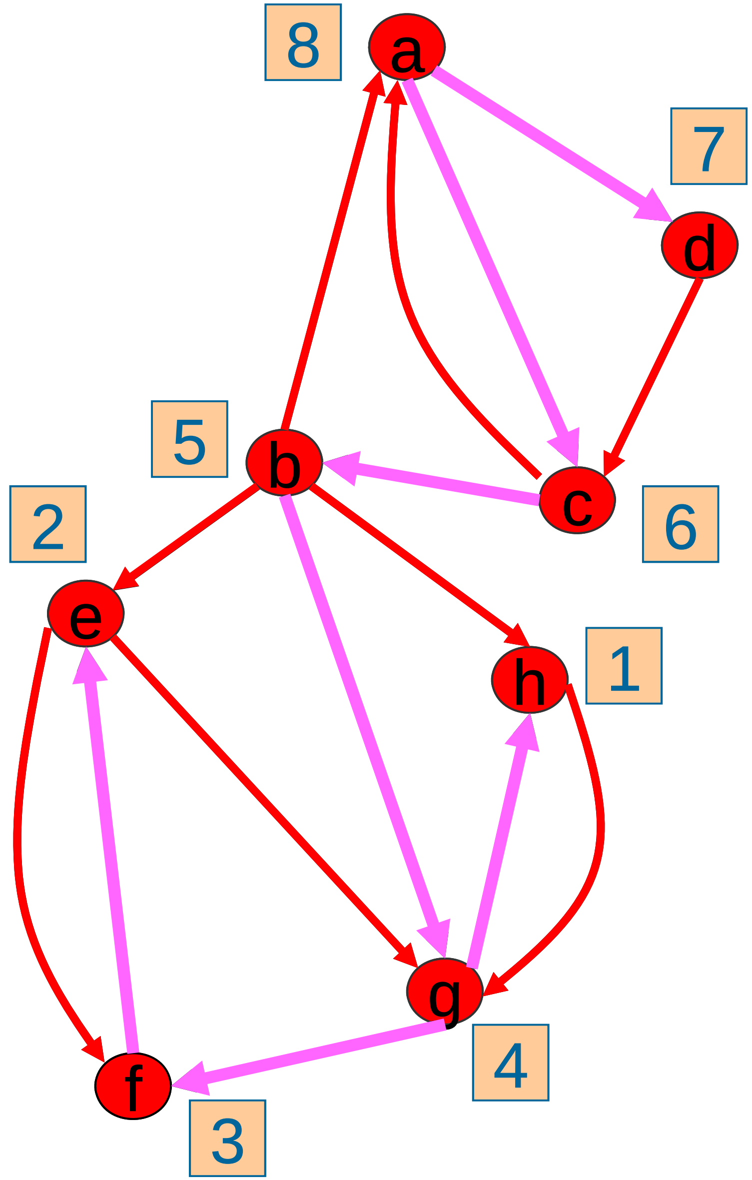
枝が白色ならば，黒く塗る

v が白色ならば

DFS-VISIT(v) を再帰呼び出し

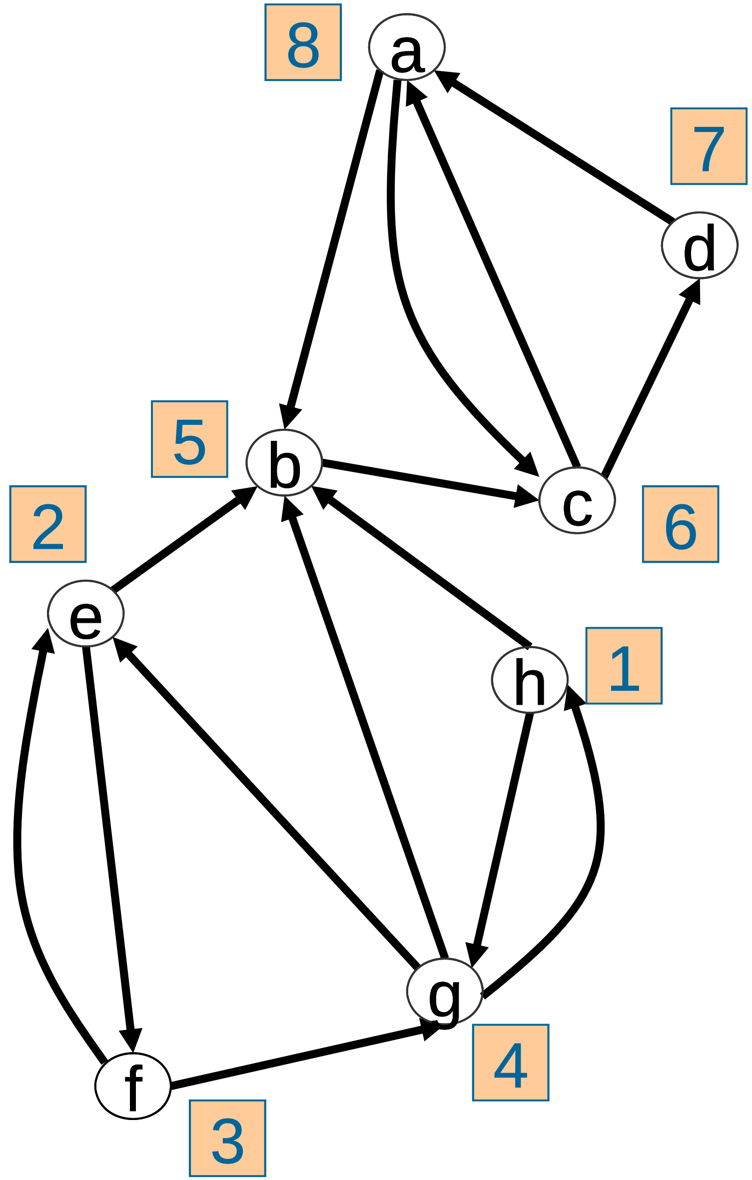
強連結成分分解の計算：例

- 1回目：普通に深さ優先探索を行い，各頂点uに対して最後に訪問した時間の順に番号を付ける←f(u)とする



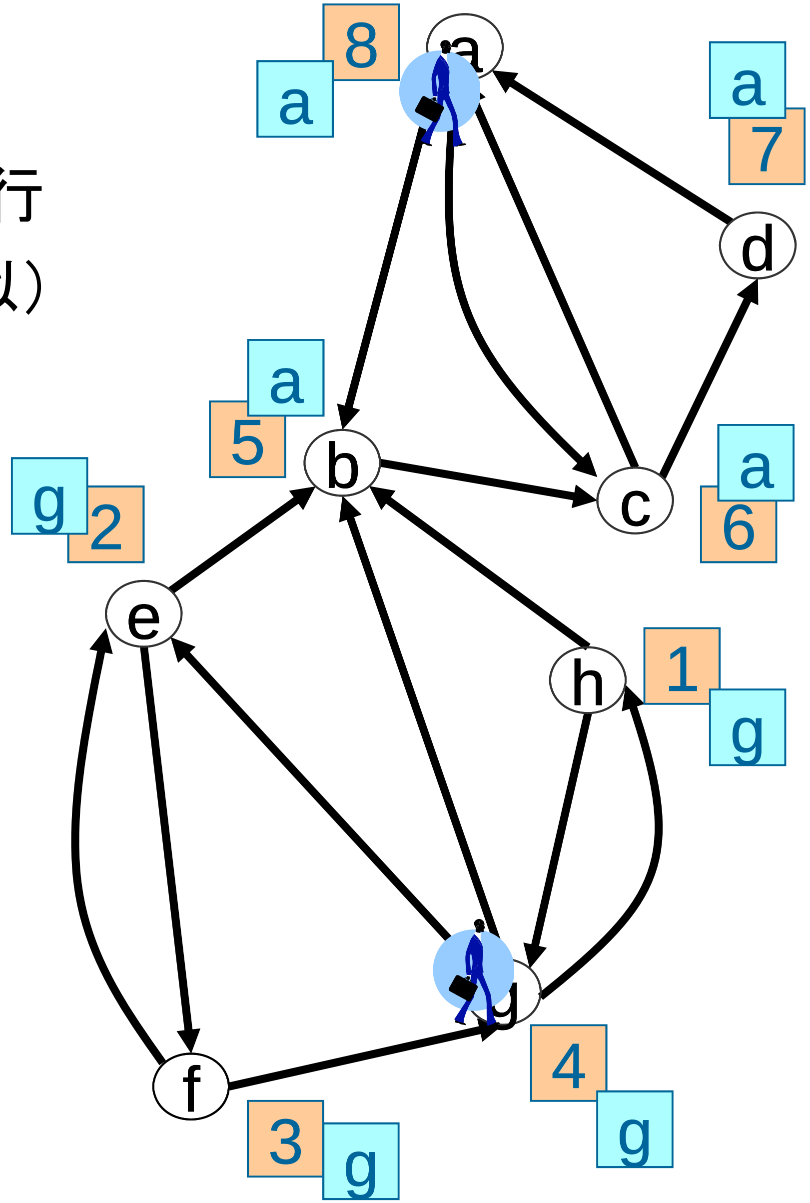
強連結成分分解の計算：例

- 2回目：
  - グラフの枝を全て逆向きにする



強連結成分分解の計算：例

- 2回目：
  - 右の深さ優先探索（修正版）を実行
  （無向グラフの連結成分分解と類似）

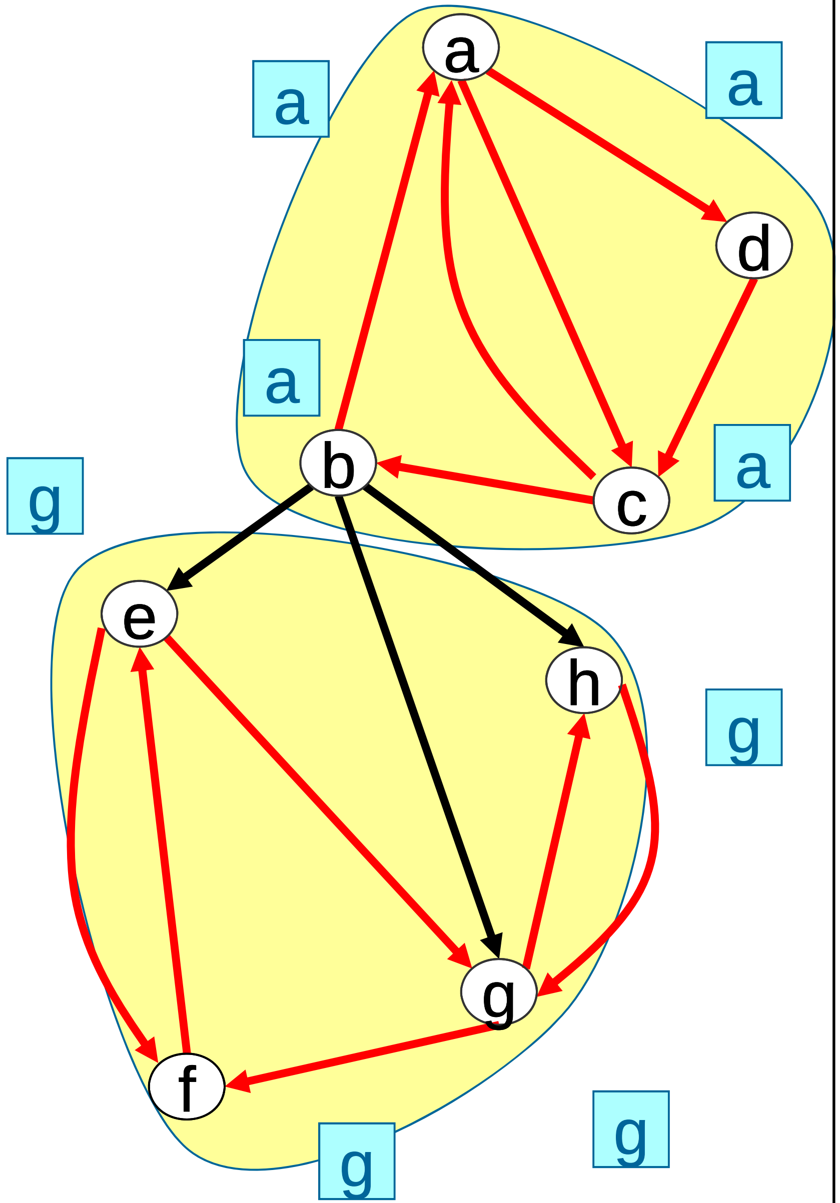


- 各頂点，各枝を白く塗る
- 各頂点 u∈V に対し，f(u)の大きい順に以下を実行
  u が白色（未走査）ならば
  k=u とおき，
  手続きDFS-VISIT(u)を実行

手続き DFS-VISIT(u)
(a) u を黒く塗り，ラベルkを付ける
(b) u に接続する各枝 (u, v) に対し，以下を実行：
  枝が白色ならば，黒く塗る
  v が白色ならば
  DFS-VISIT(v) を再帰呼び出し

強連結成分分解の計算：例

- 2回目：
  - 同じラベルごとに分解すると，強連結成分になっている



- 深さ優先探索を2回
➔時間計算量はO(m＋n)
- 正当性の証明は省略

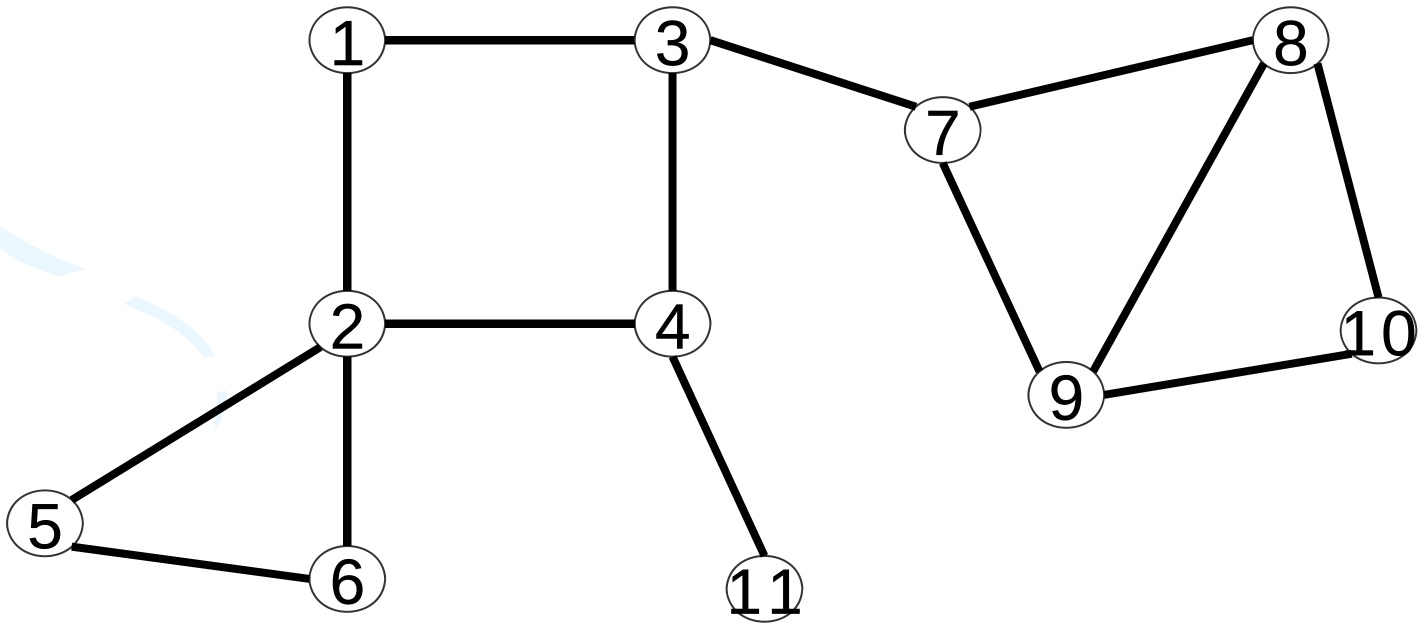
レポート問題(今回で最後)

問1:次の無向グラフに関して，下記の問に答えよ
(結果のみ書けば良い).

(1)各頂点 u に対して lowpt[u]の値を計算せよ.

(2)関節点を全て求めよ.

(3)このグラフを2連結成分に分解せよ.



レポート問題（今回で最後）

問2：次の有向グラフに関して，深さ優先探索を行うとする
（結果のみ書けば良い）
(1)初めて訪問した時間の順に，各頂点に番号を付けよ.
(2)最後に訪問した時間の順に，各頂点に番号を付けよ.
(3)このグラフを強連結成分に分解せよ.