

ACPC2017 Day 2

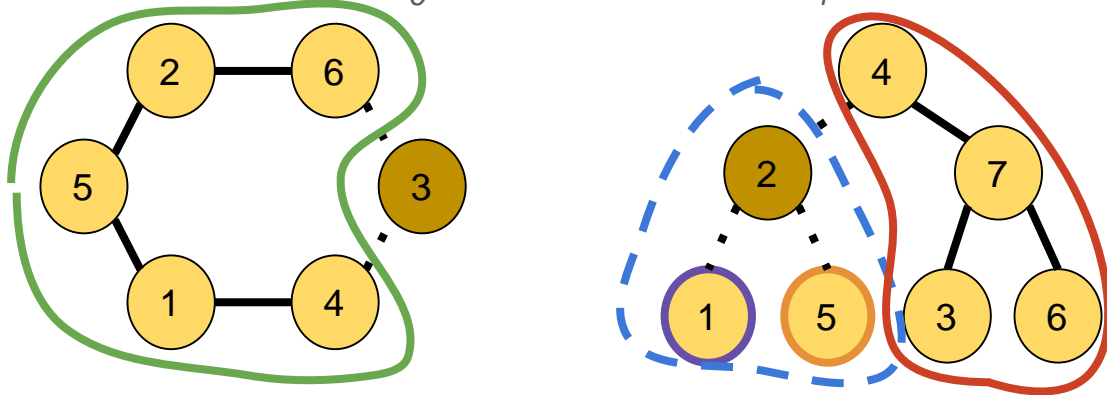
J: Cluster Network

原案・解説 : ukuku09

テスター : dohatsu・c7c7・shot

考察

- 与えられるグラフが単純な場合を考える
- 単純閉路（グラフが**二重連結**）：全体の総和 - w_i
- 木：簡単な**木DP**で計算可能
 - 頂点 i の親 p を含む連結成分と各子 v_k を含む連結成分に分断される
 - $\max\{p$ を含む連結成分, v_0 を含む連結成分, v_1 を含む連結成分, $\dots\}$



二重連結：グラフのどの頂点を取り除いてもグラフが連結

考察

- 木以外でグラフが二重連結でない場合は？

→グラフの性質を利用して木の場合に落としこもう！

関節点

- 頂点 i が**関節点**=グラフから頂点 i を取り除くと、グラフが非連結になる
- グラフが木のとき、任意の頂点が関節点

- 取り除く頂点が関節点でないなら「**全体の総和 - w_i** 」が答え
- 関節点なら、 **$\max\{p$ を含む連結成分, v_0 を含む連結成分, v_1 を含む連結成分, ...}**

木と同じになった！！

→でも、各連結の重みはどうやって計算すればいい？

解法：木DP+Lowlink

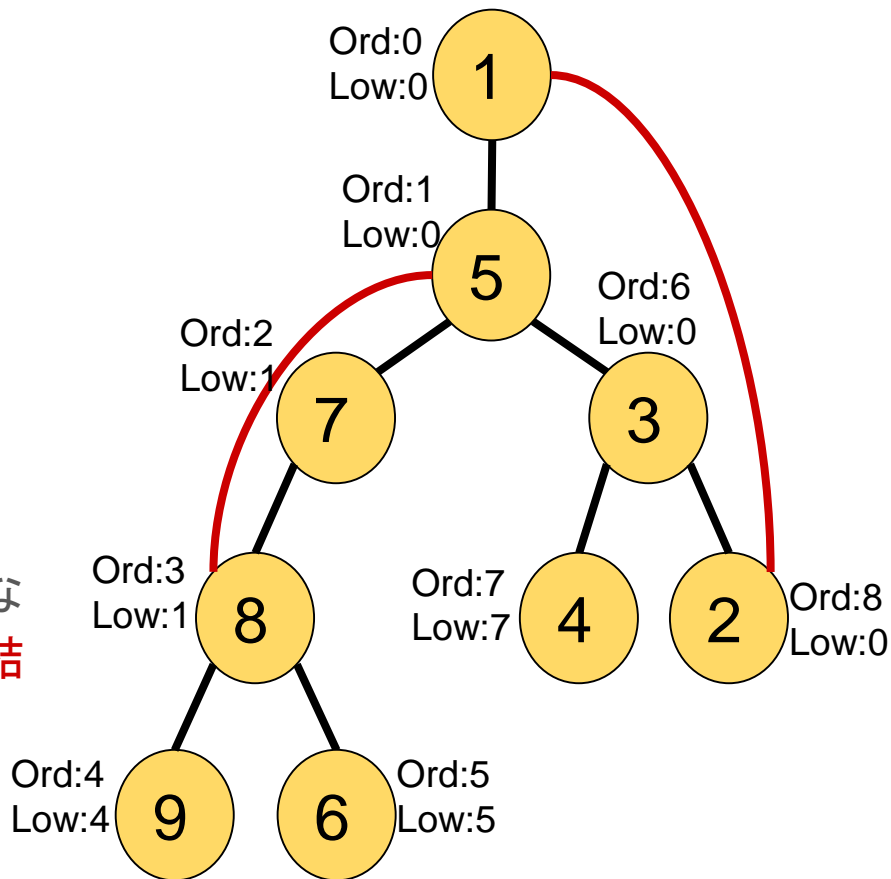
- 関節点列挙に使われるLowlinkの考え方を応用する
- Lowlinkってなに...？
 - グラフの適当な頂点を根として深さ優先探索（DFS）
 - DFSで通過した頂点と辺のみを使った木=DFS Tree
 - このとき、DFS Treeに使われなかった辺=後退辺
 - 頂点 v から、次の移動のみを許して到達できる任意の頂点のうち、DFSの訪問順序が最小である頂点 u の訪問順序を、頂点 v のLowlinkという
 - DFS Treeの葉に向かう辺を任意の回数使用
 - 後退辺をたかだか1回まで使用

Lowlink

- Ord : 訪問順序
- Low : Lowlink

Lowlinkが求まる

→DFS木において祖先であるような
関節点を取り除いたとき、**どの連結
成分に属するかがわかる**



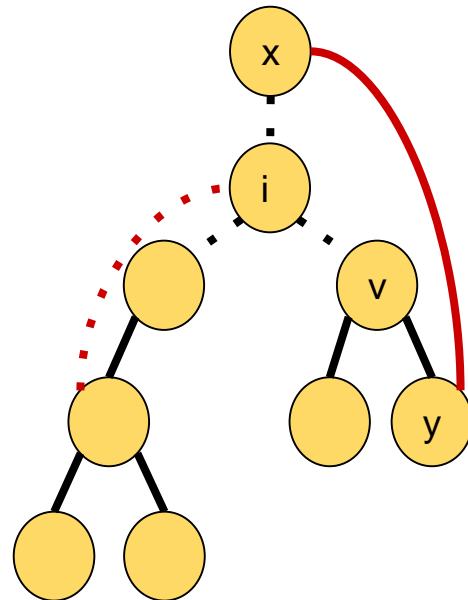
解法：木DP+Lowlink

- 頂点*i*について、その祖先*x*と子孫*y*が**後退辺**でつながっているならば、頂点*i*の子*v*を根とし子孫*y*を含む部分木は、元のグラフで頂点*i*を取り除いてもその祖先の連結成分と連結

→実は、そのような子*v*について

頂点*v*のLowlink < 頂点*i*の訪問順序

が成り立つ！



Lowlink

- 取り除く頂点が関節点でないなら「全体の総和 - w_i 」が答え
- 関節点なら、 $\max\{p$ を含む連結成分, v_0 を含む連結成分, v_1 を含む連結成分, ...}

p を含む連結成分の重み

=全体の総和 - ($dp[i] - \sum\{dp[v] \mid \text{頂点 } v \text{のLowlink} < \text{頂点 } i \text{の訪問順序}\}$)

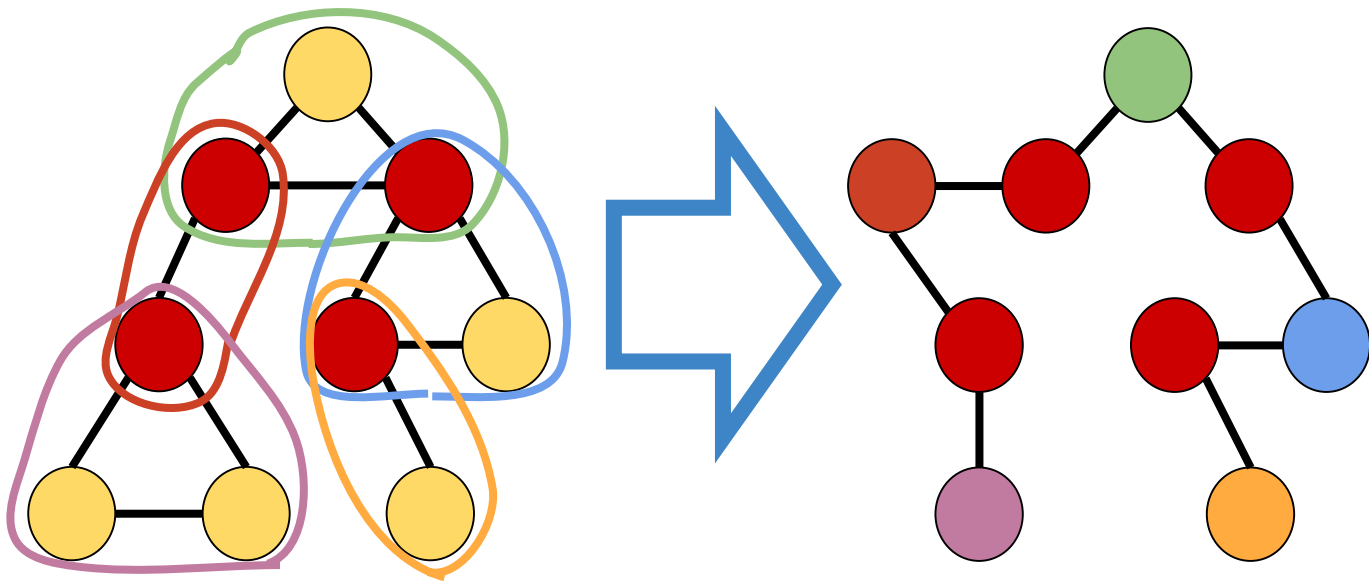
- $dp[i]$:=DFS木における、頂点 i の部分木の重み

v_k を含む連結成分の重みは、 $dp[v_k]$ のままでもOK

- Lowlinkの計算 : $O(N+M)$
- 木DP : $O(N+M)$

別解：Block Cut Tree+木DP

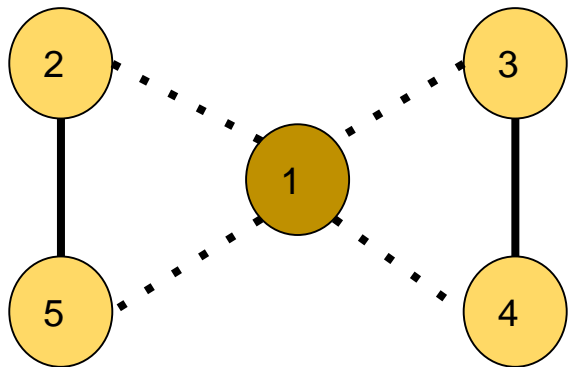
- 関節点列挙のアルゴリズムを応用→二重頂点連結成分分解
 - 二重頂点連結成分=どの頂点を取り除いても連結な成分
- さらにBlock Cut Treeなる木をつくると、木DPの結果がそのまま使える



注意

- 橋の列挙や二重辺連結成分分解ではダメ！！

こういうケースで落ちる↓（作問者とテスターがハマりました...）



橋はないけど頂点1を取り除くと
グラフが二つに分断される

講評

- First Accepted:
 - On-site: いません。（これだから下等生物は困るわよね～）
 - Online: latte0119 (3h30m)

- Success Rate: 50.00% (2/4)

ジャツジ解

- dohatsu (lowlink) 133行 2107B
- c7c7 (lowlink) 39行 974B
- shot (lowlink) 86行 1762B
- uku (lowlink) 125行 2483B
- uku (block cut tree) 150行 2963B