RUPC2018 Day2
L: Tree Fragments

原案・解説：ukuku09

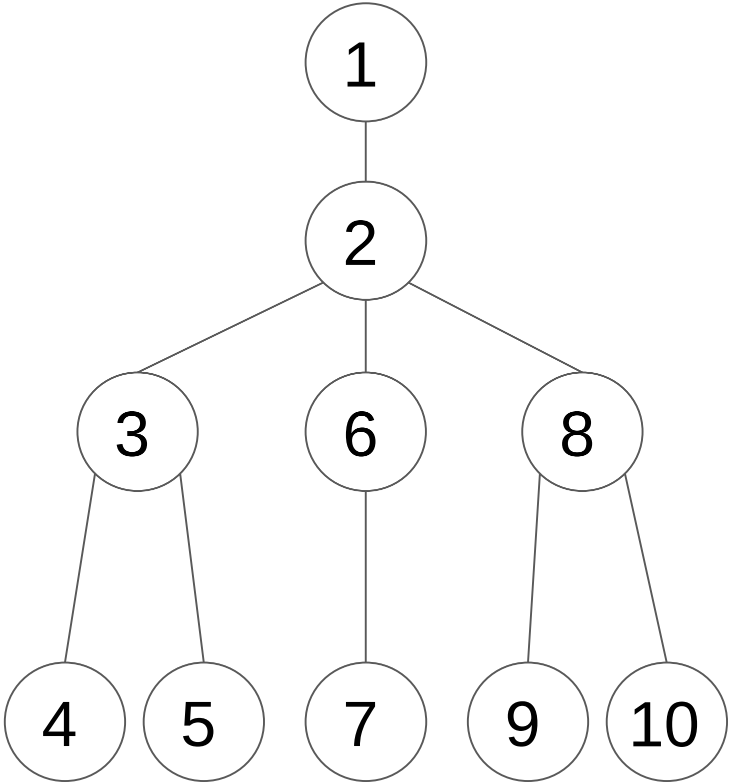
問題概要

- 頂点数Nの重み付き木Tが与えられる
- 以下のクエリがQ回与えられるので答えよ

  a_i-b_iパス上の頂点と、それに接続する辺を取り除いてできる連結成分の重みのうち、最大値を求める

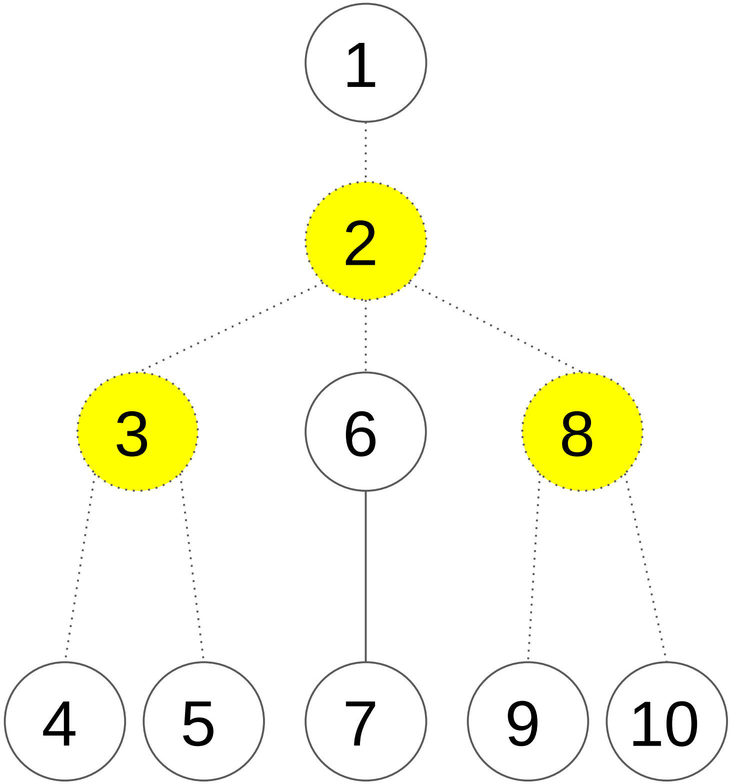
- N,Q<=10^5

Sample Input 1



Sample Input 1

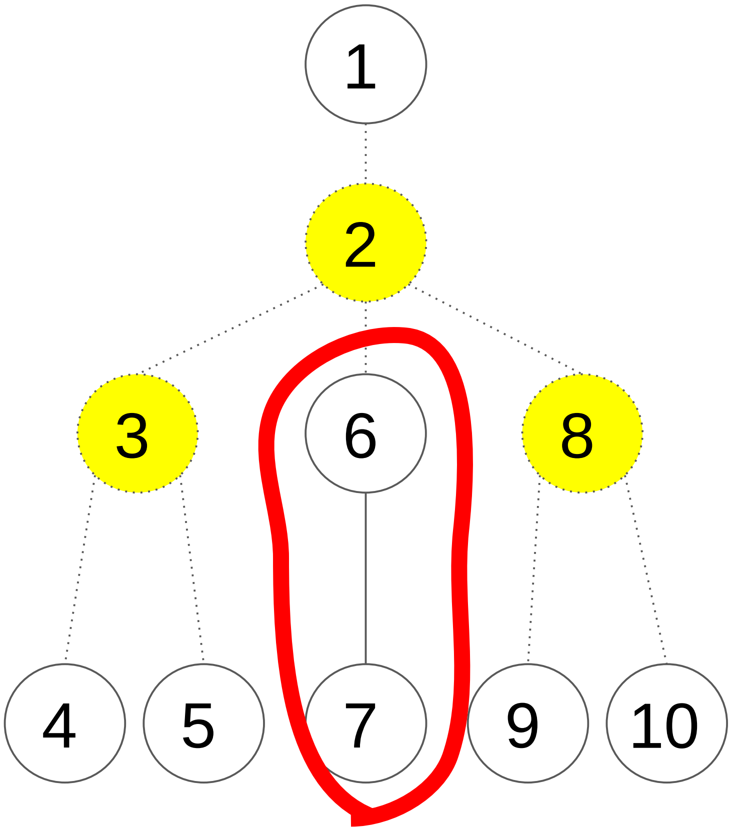
ここを消すと…



Sample Input 1

ここを消すと…

答えは6+7=13



解法

- ダブリング O(N+QlogN)
  - 簡単な木DPとLCAの知識も必要
  - 適当な根付き木にして、木のパス上のRMQに帰着させる
  - クエリごとに、O(logN)で計算できる

解法

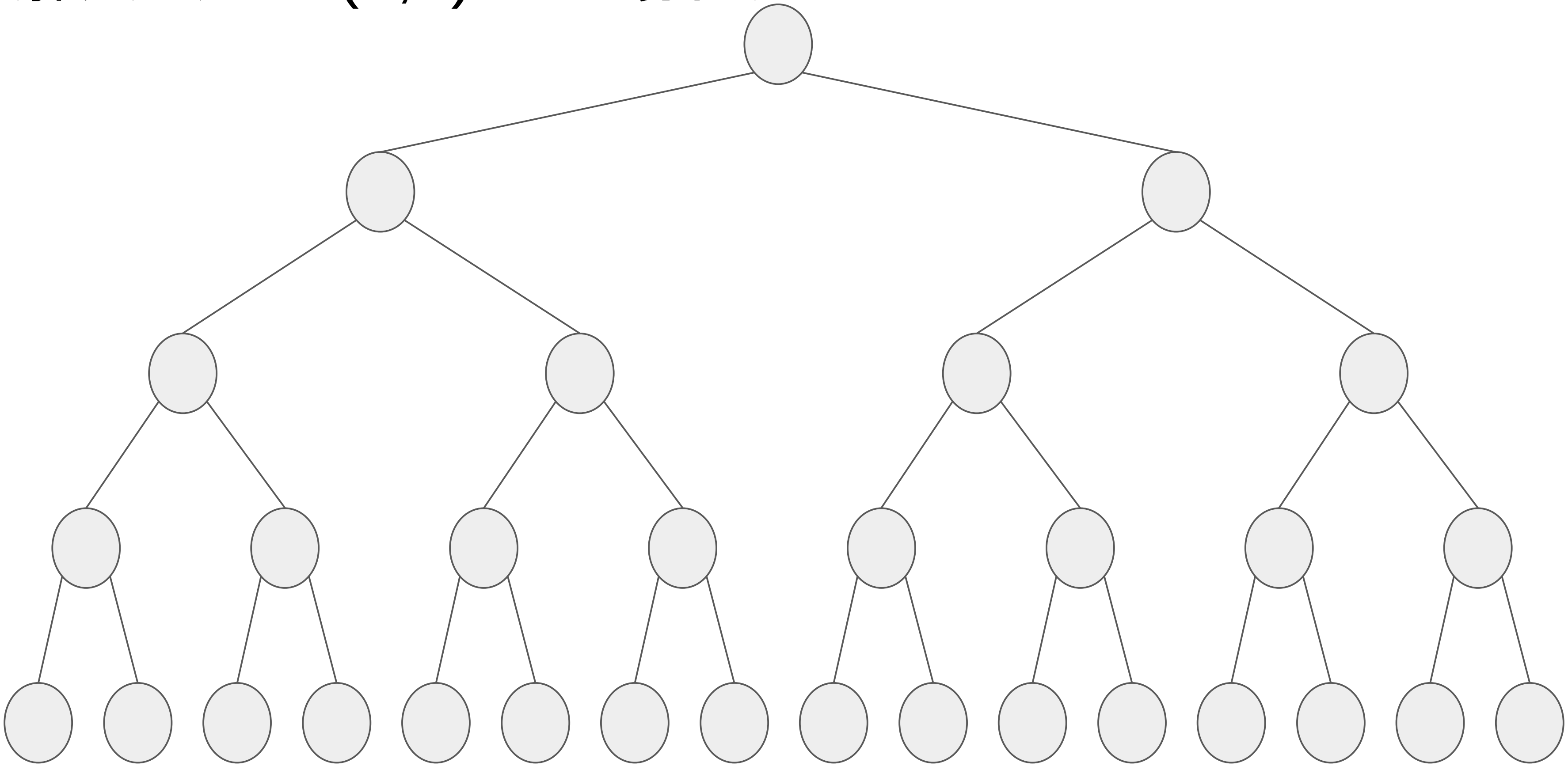
- 説明を簡単にするため、二分木の場合を例に考えてみる
- 決してスライドに一般の木を描くのが面倒臭かったからではない
- 決してない

ここで、

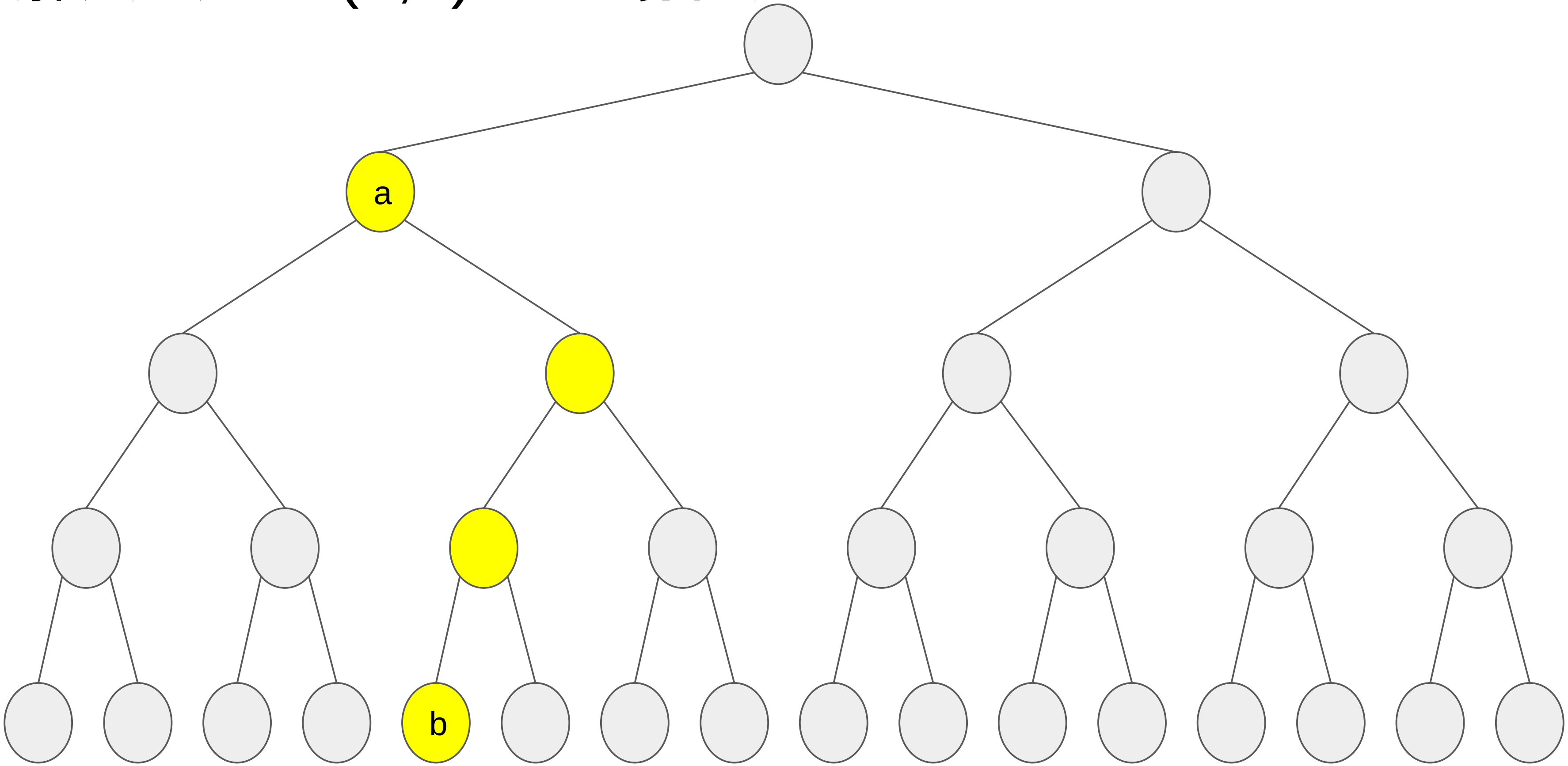
- クエリ(a,b)：頂点aの深さ≤頂点bの深さ（一般性を失わない）
- T_v：＝頂点vを根とする部分木

としておく。

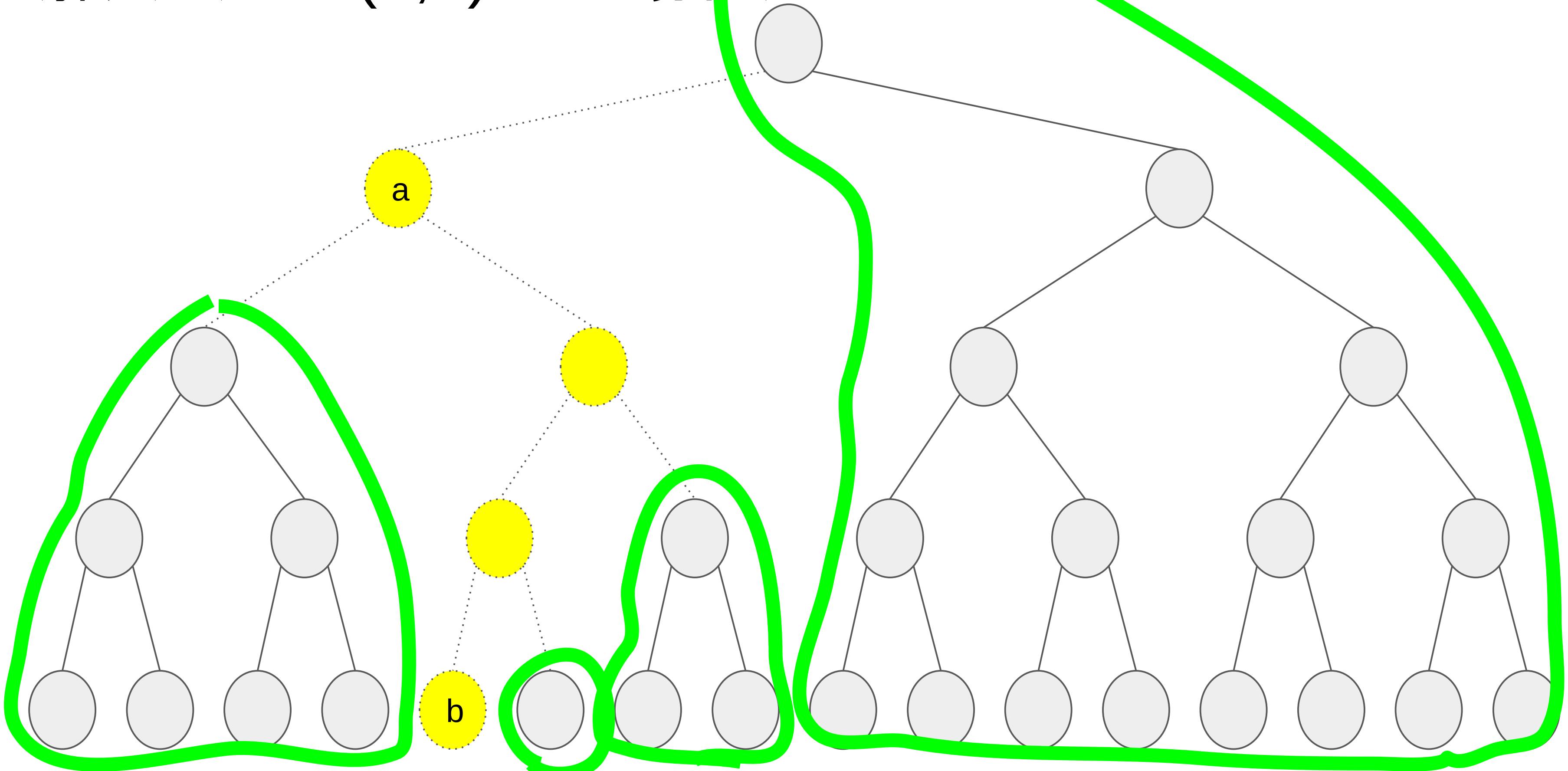
解法（LCA(a,b)=aの場合）



解法（LCA(a,b)=aの場合）

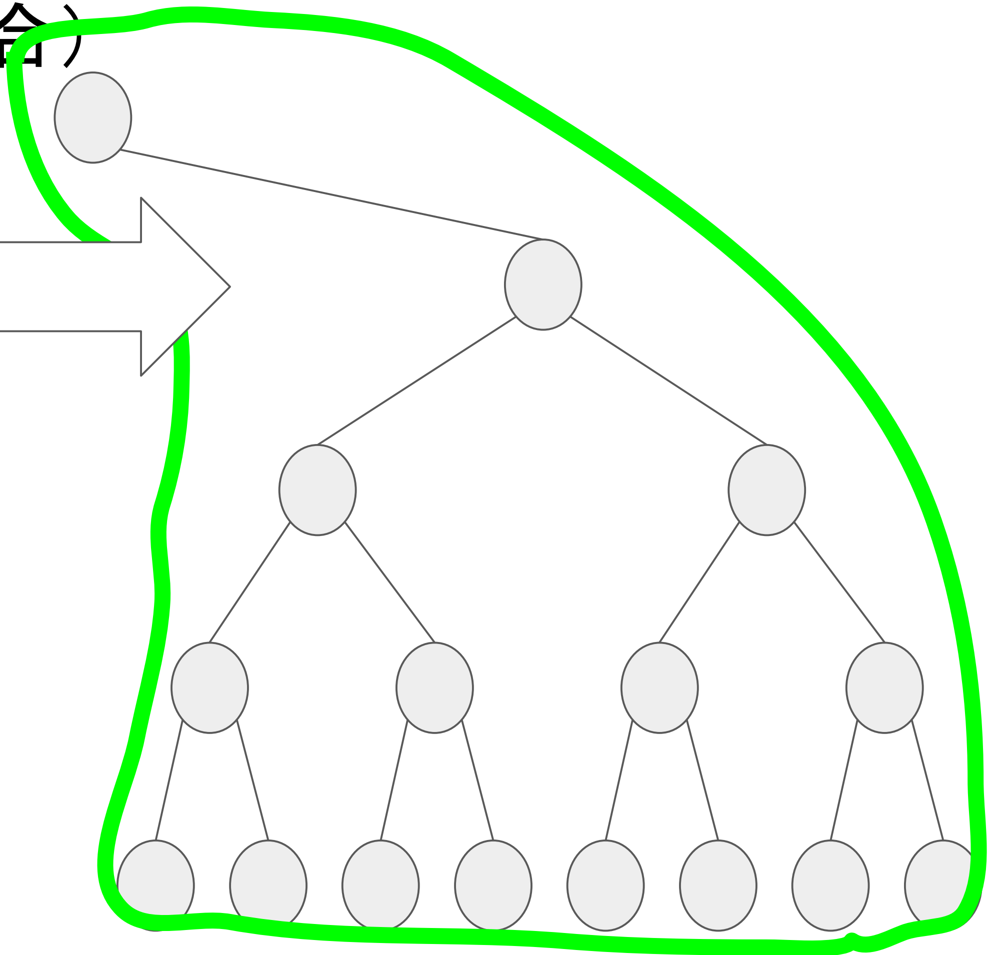


解法（LCA(a,b)=aの場合）

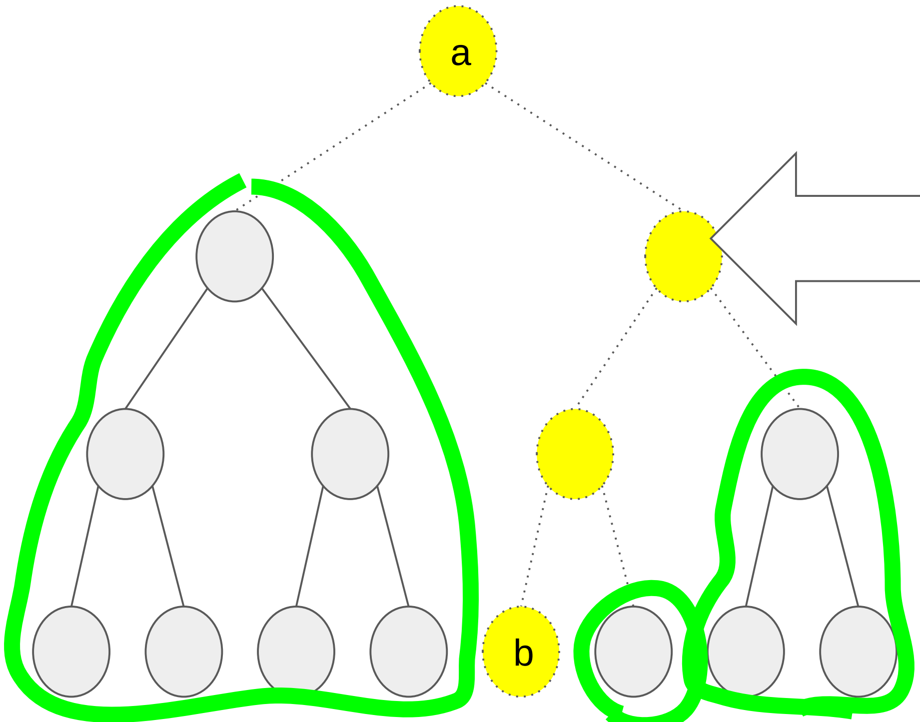


解法（LCA(a,b)=aの場合）

こっちは簡単。
重みの総和ーT_aの重み



解法（LCA(a,b)=aの場合）



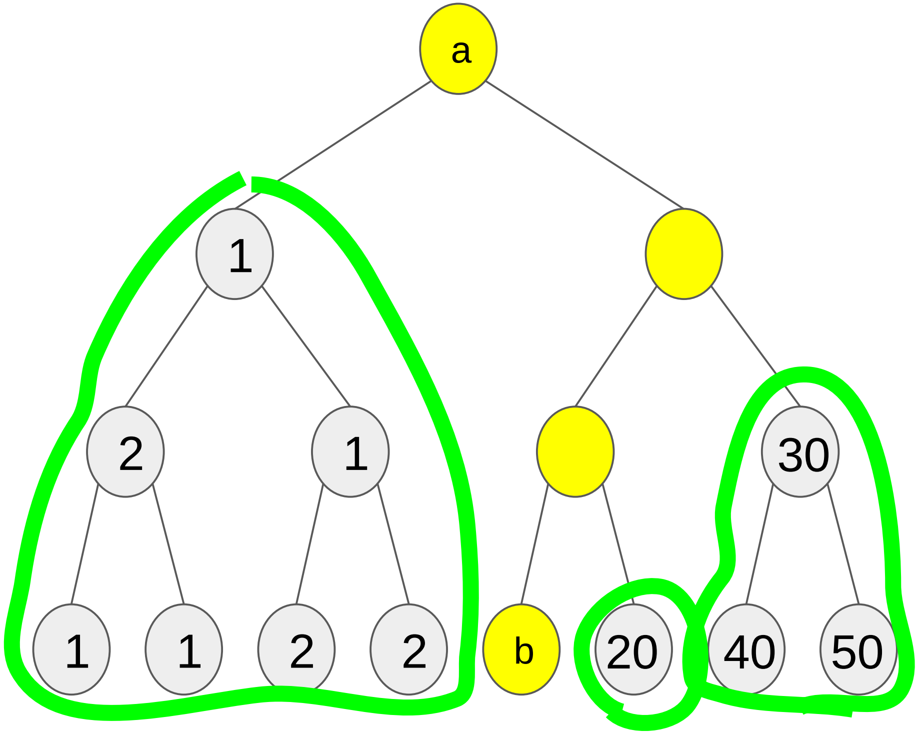
こっちに含まれる連結成分の重みの最大値を求めよう（部分問題1）

解法（LCA(a,b)=aの場合）

パス上の各頂点vに、

f(v)：＝T_uの重み（uはvの兄弟）

を持たせてみると…

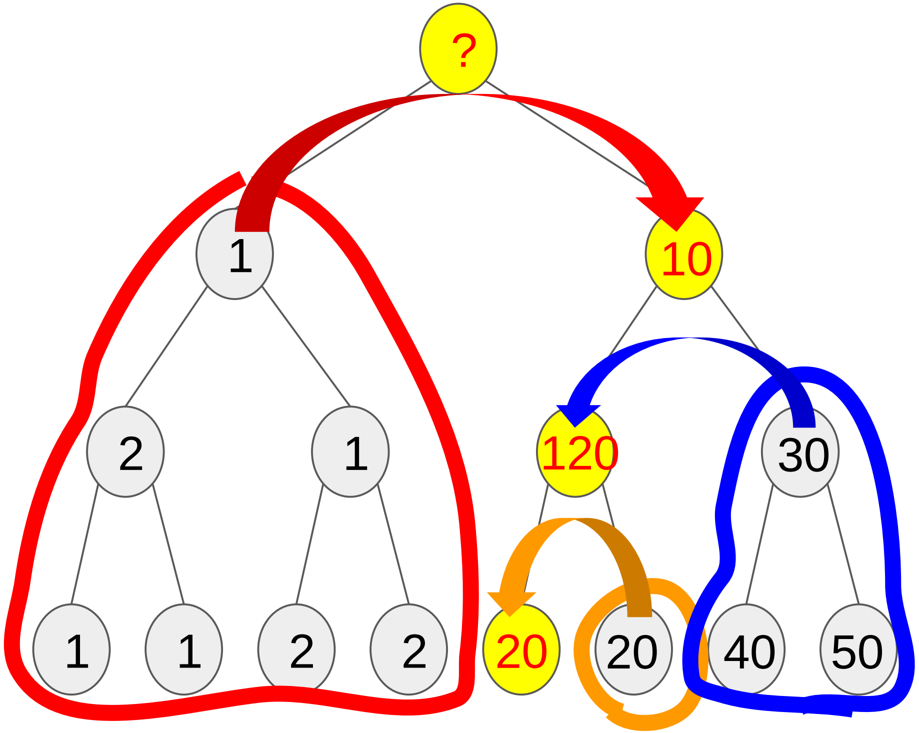


解法（LCA(a,b)=aの場合）

パス上の各頂点vに、

f(v)：＝T_uの重み（uはvの兄弟）

を持たせてみると…



解法（LCA(a,b)=aの場合）

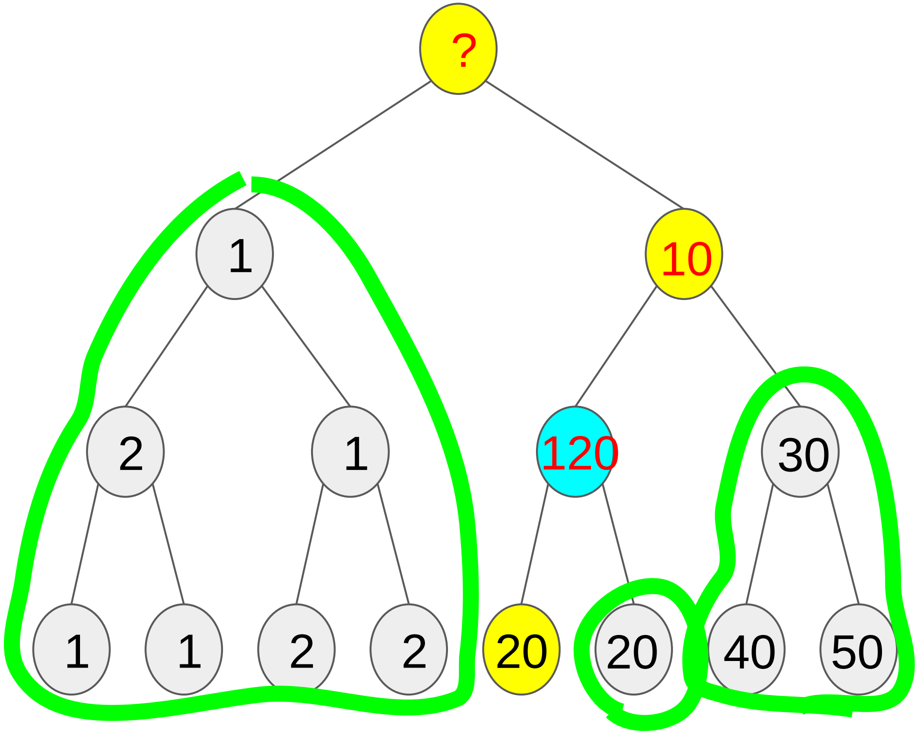
パス上の各頂点vに、

f(v)：＝T_uの重み（uはvの兄弟）

を持たせてみると…

パス上のf(v)の最大値が、部分問題1の答えになっている！！（※aの兄弟はT_a上にないので、f(a)は見ない）

- f(v)の定義より、明らかにパス上のf(v)はパス上の頂点の重みを含まない。



解法（LCA(a,b)=aの場合）

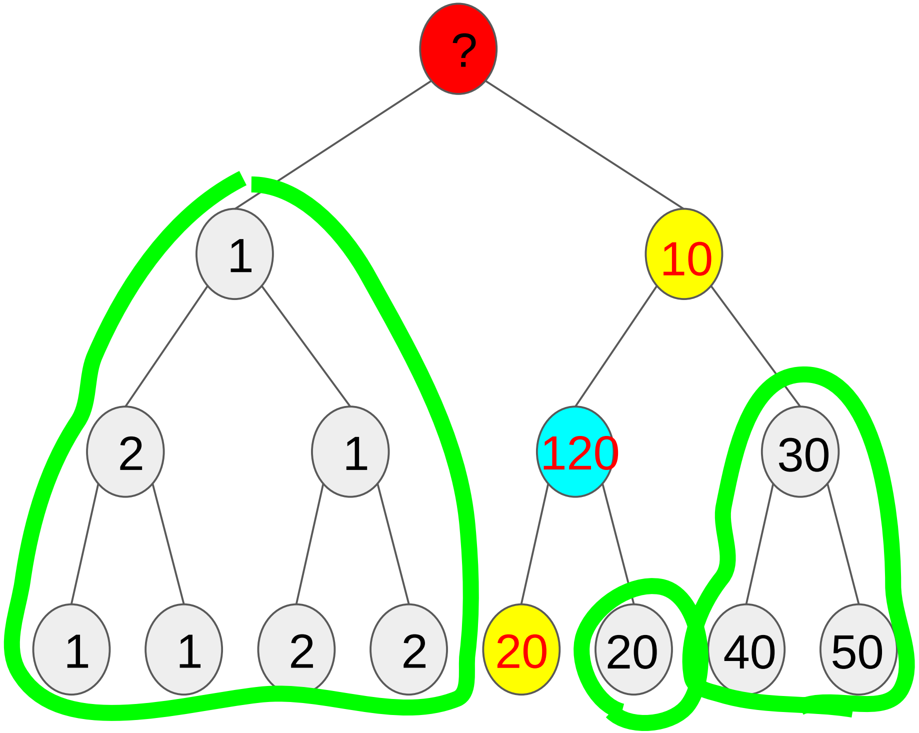
パス上の各頂点vに、

f(v)：＝T_uの重み（uはvの兄弟）

を持たせてみると…

パス上のf(v)の最大値が、部分問題1の答えになっている！！（※aの兄弟はT_a上にないので、f(a)は見ない）

- f(v)の定義より、明らかにパス上のf(v)はパス上の頂点の重みを含まない。



解法（LCA(a,b)=aの場合）

- f(v)：＝T_uの重み（uはvの兄弟）はクエリに寄らない➡前処理可
- 一般に、f(v)：＝T_{u_i}の重みの最大値（u_iはvの兄弟）とすれば良い
- 変更クエリがないので、この木上のRange Maximum Queryはダブリングで求められる
- max{総和ーT_aの重み,パス上のf(v)の最大値,
  T_{c_i}の重みの最大値（c_iはbの子）}

解法（LCA(a,b)=aの場合）

パス上の各頂点vに、

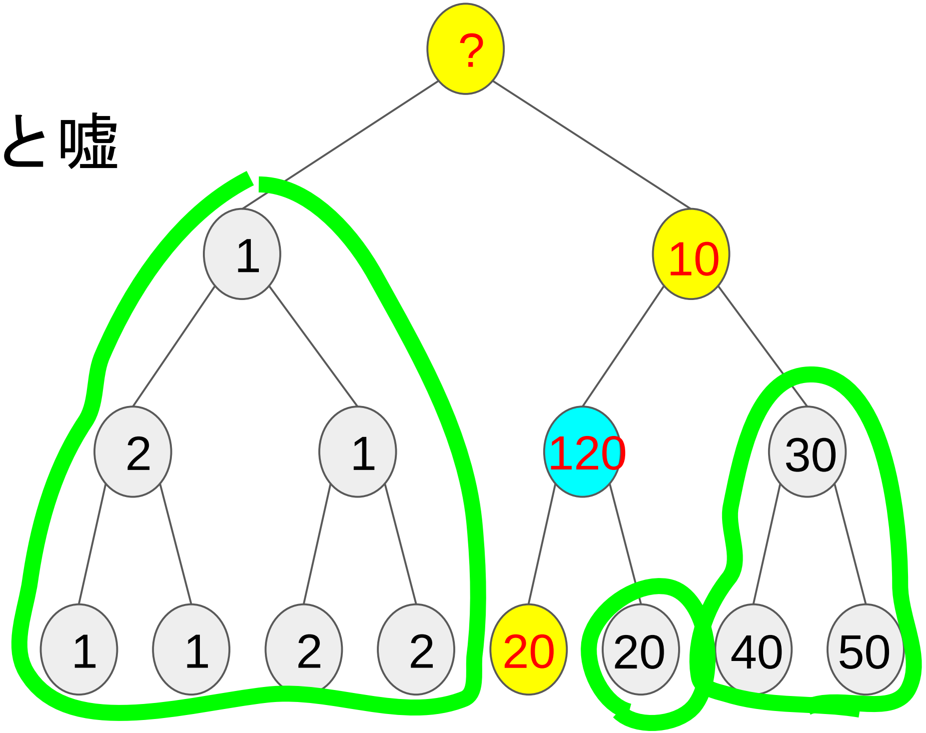
f(v)：＝T_uの重み（uはvの兄弟）

を持たせてみると…

ちょっと嘘

<u>パス上のf(v)の最大値が、部分問題1の答えになっている</u>！！（※aの兄弟はT_a上にないので、f(a)は見ない）

- f(v)の定義より、明らかにパス上のf(v)はパス上の頂点の重みを含まない。

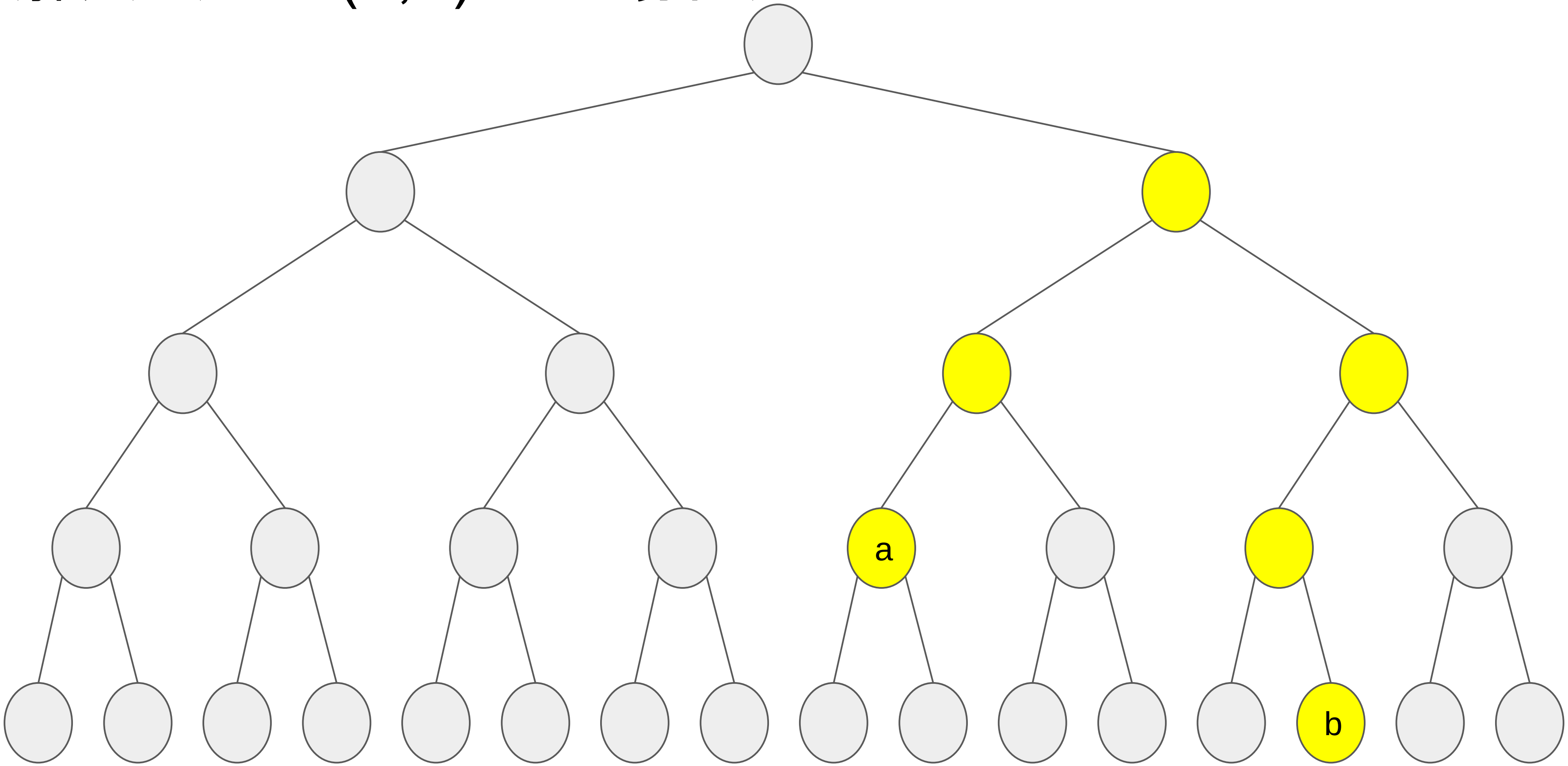


解法（LCA(a,b)=aの場合）

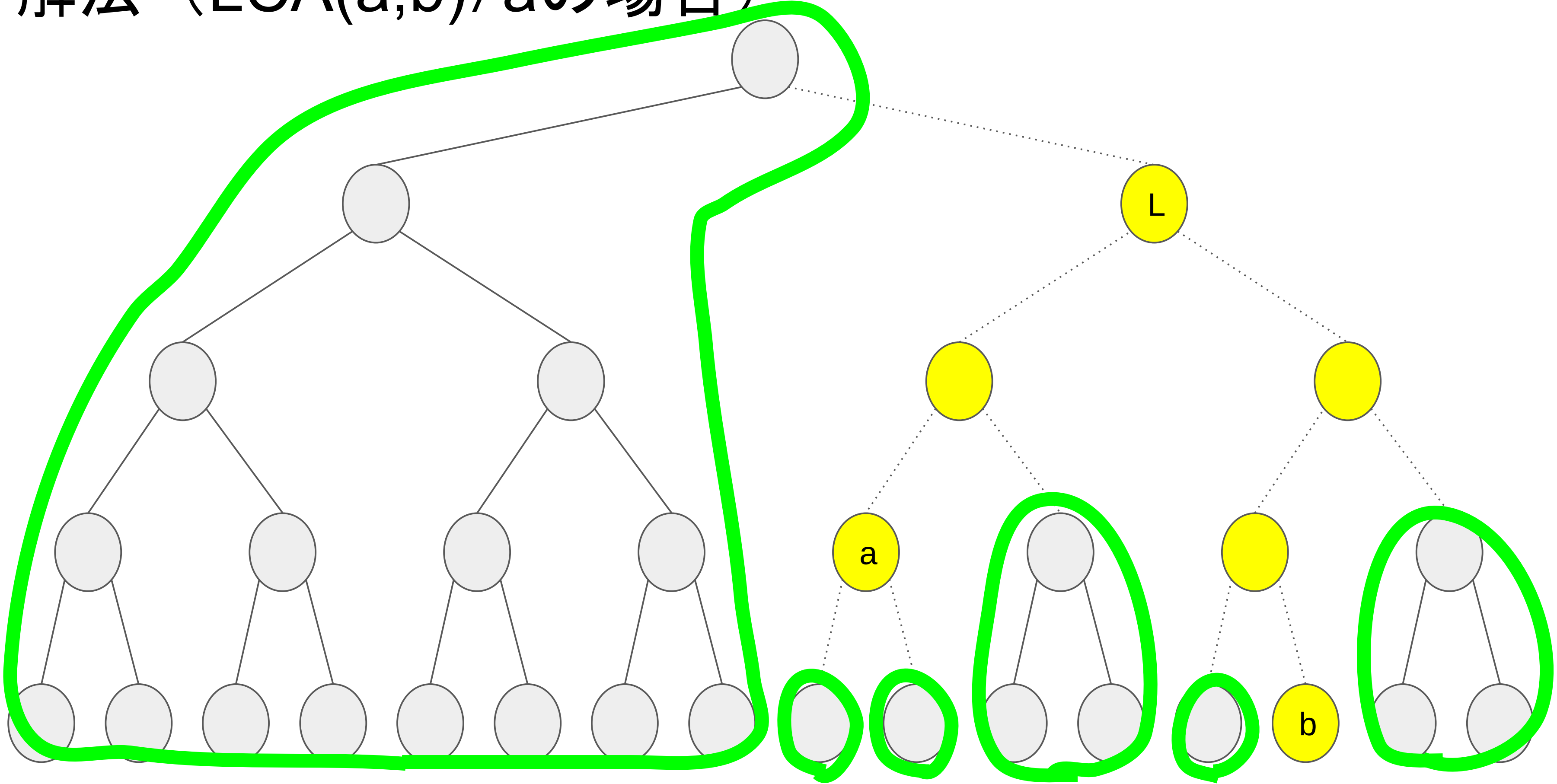
- f(v)：＝W(vの兄弟u)はクエリに寄らない➡□前処理可
- 多分木の場合、f(v)：＝T_{u_i}の重みの最大値（u_iはvの兄弟）とすれば良い
- 変更クエリがないので、この木上のRange Maximum Queryはダブリングで求められる
- max{総和ーT_aの重み,パス上のf(v)の最大値,
  T_{c_i}の重みの最大値（c_iはbの子）}

これも忘れずに！

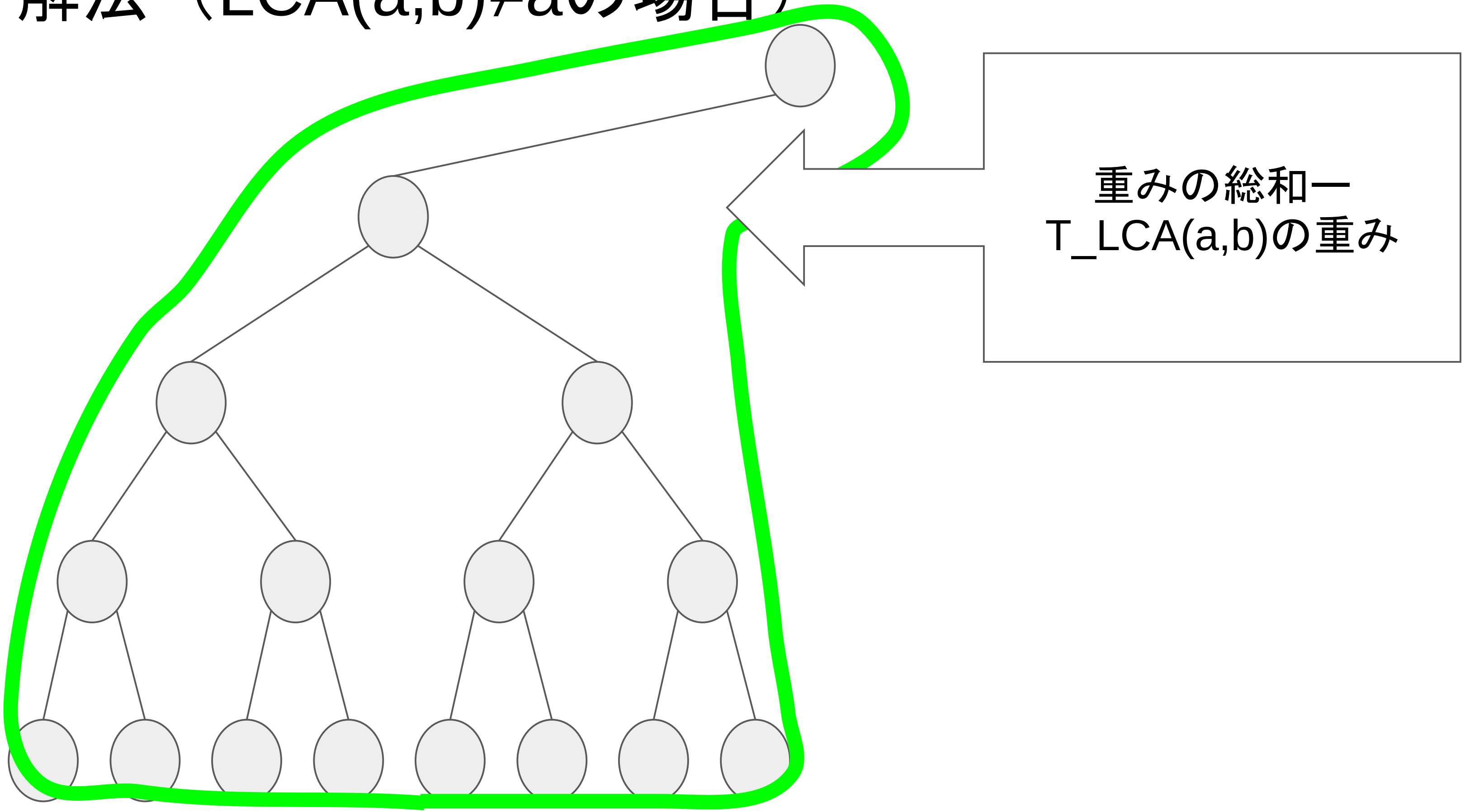
解法（LCA(a,b)≠aの場合）



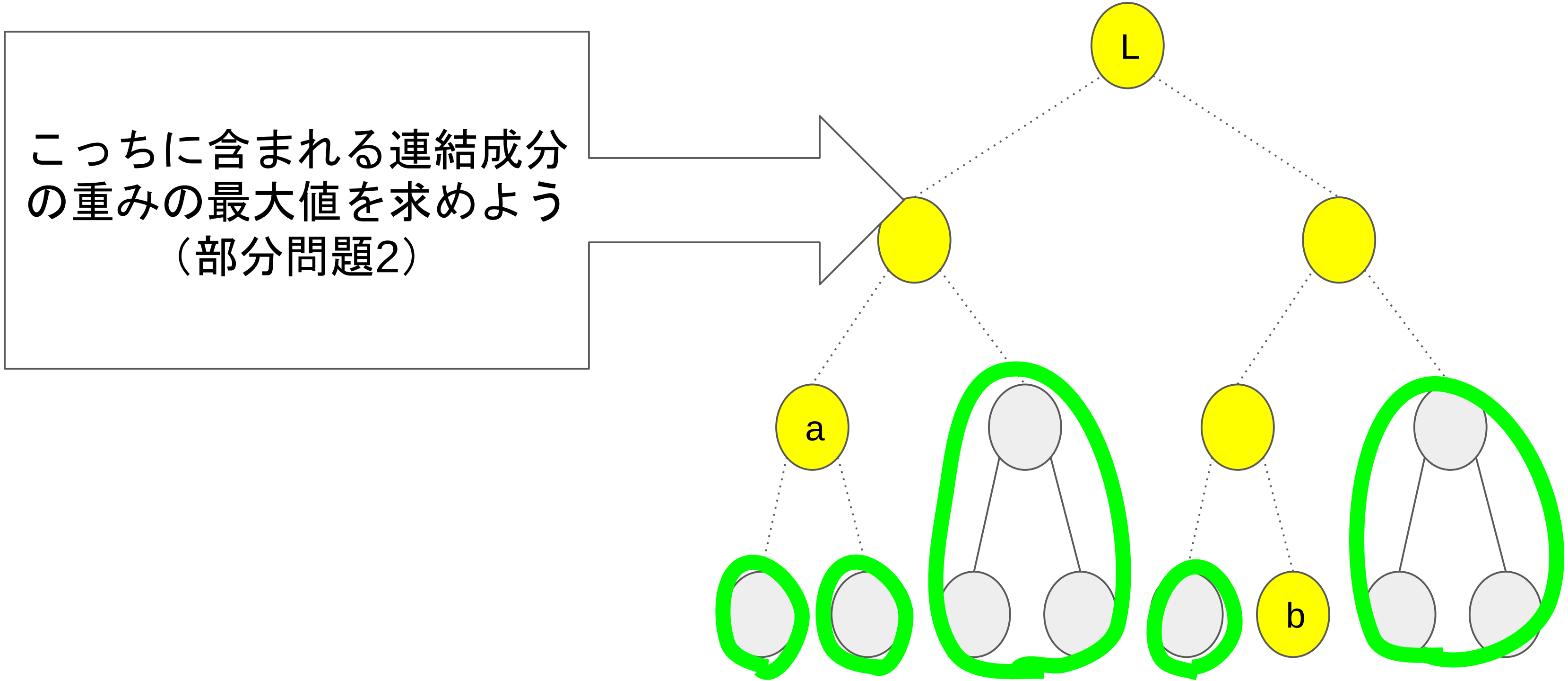
解法（LCA(a,b)≠aの場合）



解法（LCA(a,b)≠aの場合）

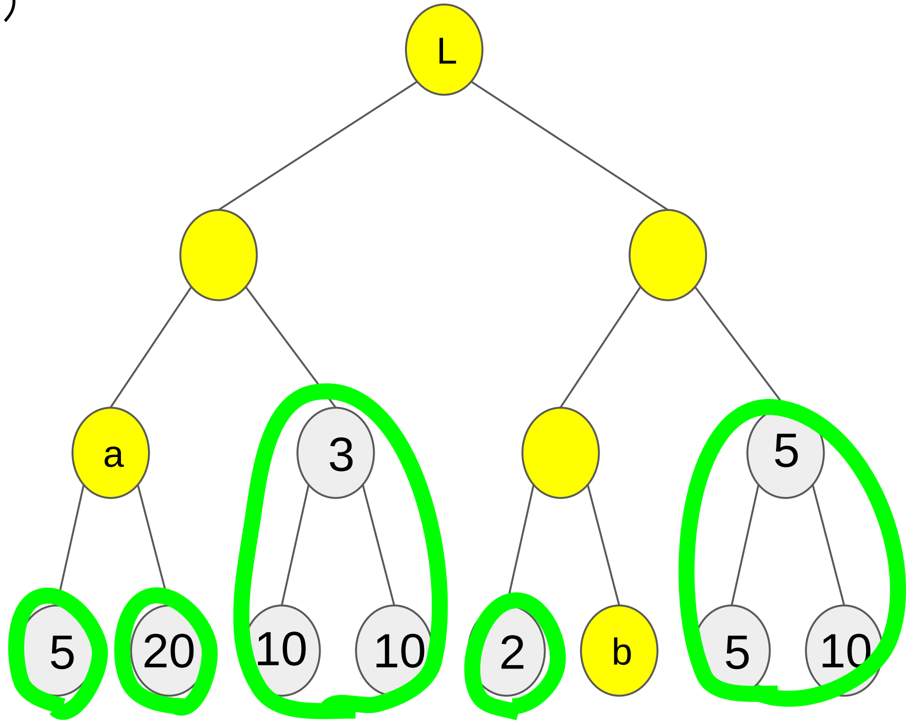


解法（LCA(a,b)≠aの場合）



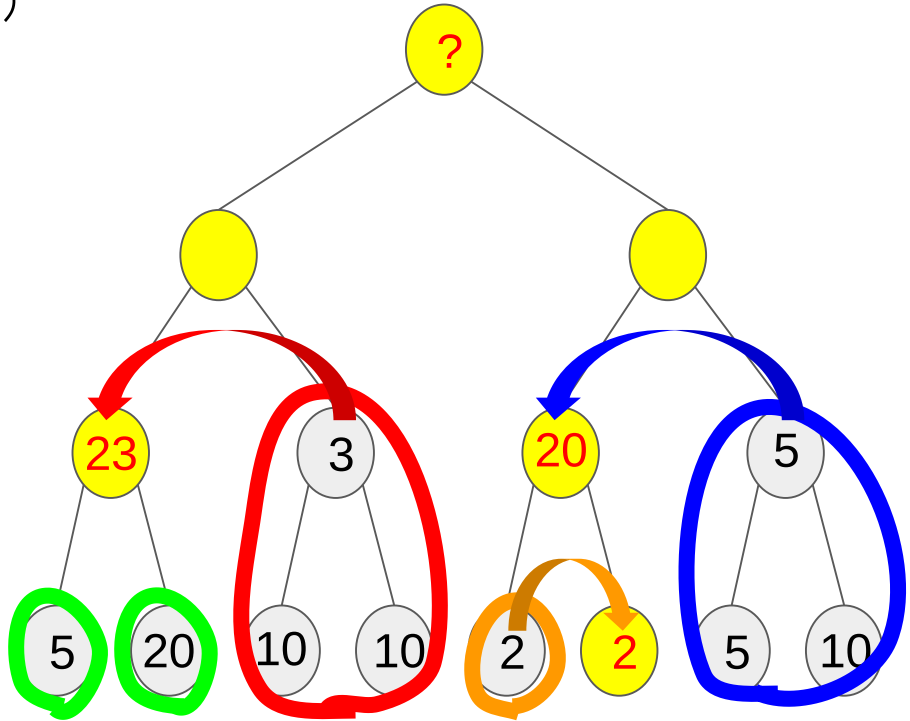
解法（LCA(a,b)≠aの場合）

- さっきと同様パス上のf(v)を求めておく
  （f(v)：＝T_uの重み（uはvの兄弟））



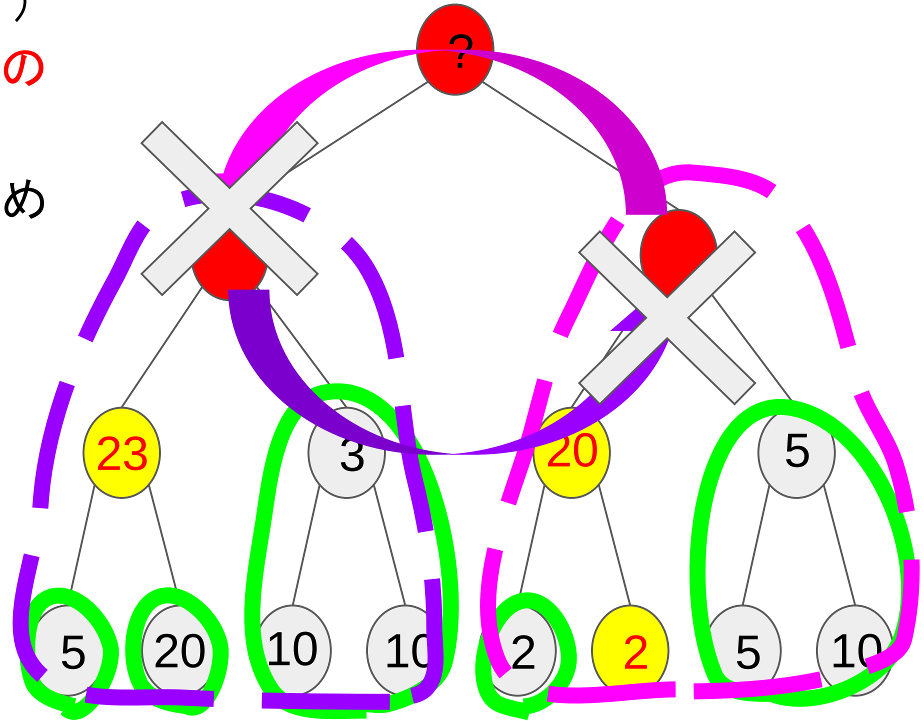
解法（LCA(a,b)≠aの場合）

- さっきと同様パス上のf(v)を求めておく
  （f(v)：＝T_uの重み（uはvの兄弟））



解法（LCA(a,b)≠aの場合）

- さっきと同様パス上のf(v)を求めておく
  （f(v)：＝T_uの重み（uはvの兄弟））
- 今度はLCA(a,b)だけでなく、その子の値も見てはいけない
  - どちらもパスに含まれているため

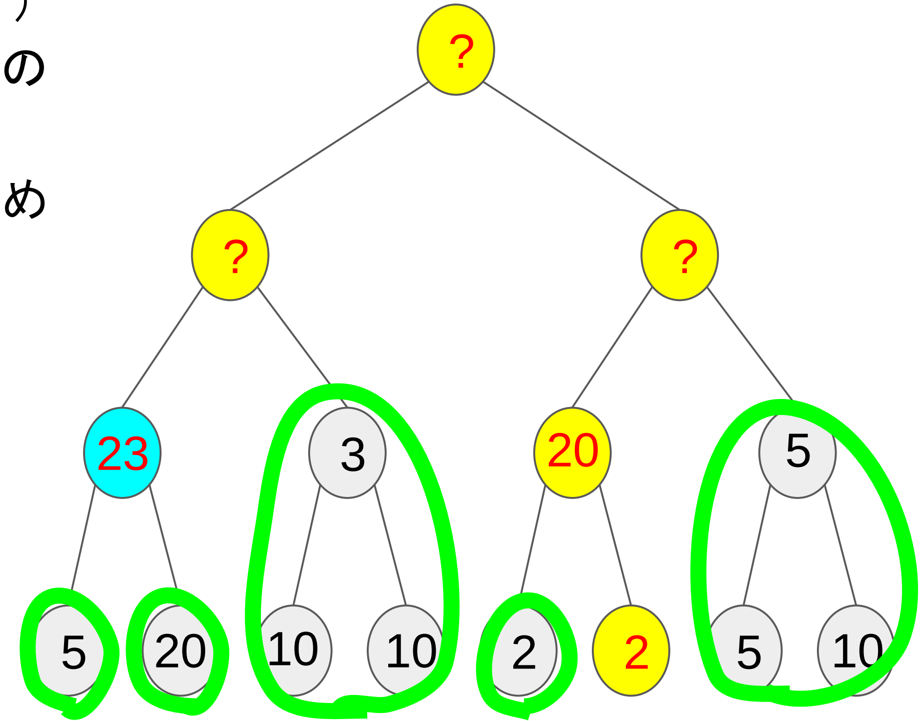


解法（LCA(a,b)≠aの場合）

- さっきと同様パス上のf(v)を求めておく
（f(v)：＝T_uの重み（uはvの兄弟））
- 今度はLCA(a,b)だけでなく、その子の値も見てはいけない
  - どちらもパスに含まれているため

それ以外のパス上のf(v)の最大値

が答え



解法（LCA(a,b)≠aの場合）

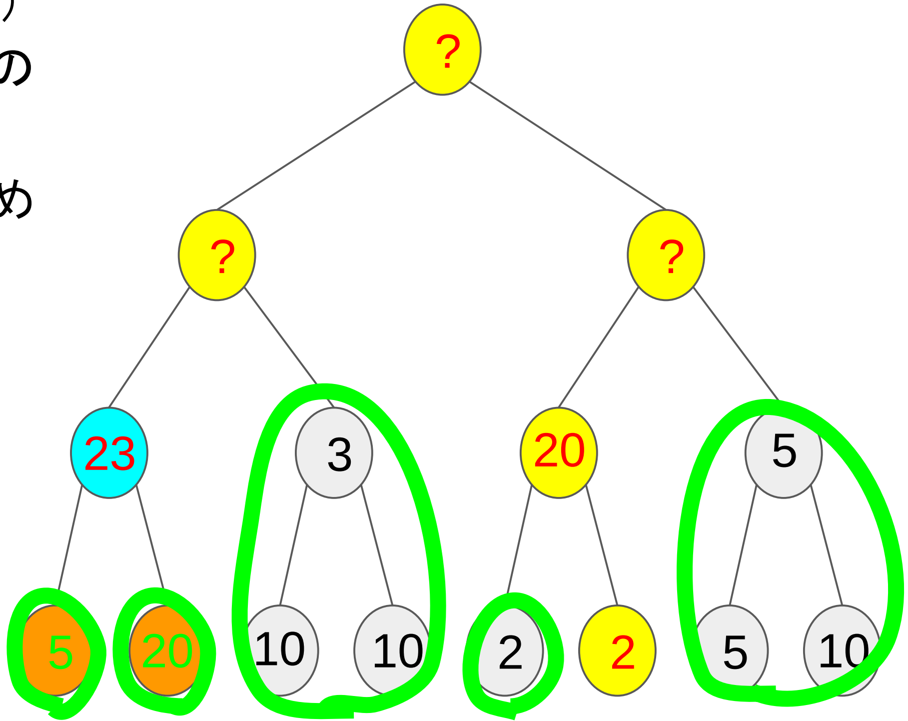
- さっきと同様パス上のf(v)を求めておく（f(v)：＝T_uの重み（uはvの兄弟））
- 今度はLCA(a,b)だけでなく、その子の値も見てはいけない
  - どちらもパスに含まれているため

それ以外のパス上のf(v)の最大値 or

T_{c_i}の重みの最大値（c_iはaの子） or

T_{d_i}の重みの最大値（d_iはbの子）

が答え



解法（LCA(a,b)≠aの場合）

- f(v)：＝T_uの重み（uはvの兄弟）はクエリに寄らない➡前処理可
- 一般に、f(v)：＝T_{u_i}の重みの最大値（u_iはvの兄弟）とすれば良い
- 変更クエリがないので、この木上のRange Maximum Queryはダブリングで求められる
- **一般に、T_{x_i}の重みの最大値（x_iはパス上にないLCA(a,b)の子）を求める必要がある（超重要！！！）←上位3つまでわかってれば良い**
- max{重みの総和ーT_LCA(a,b)の重み,パス上のf(v)の最大値,
  T_{c_i}の重みの最大値（c_iはaの子）,
  T_{d_i}の重みの最大値（d_iはbの子）,
  T_{x_i}の重みの最大値（x_iはパス上にないLCA(a,b)の子）}

解法（LCA(a,b)≠aの場合）

LCA(a,b)=aの時と一緒

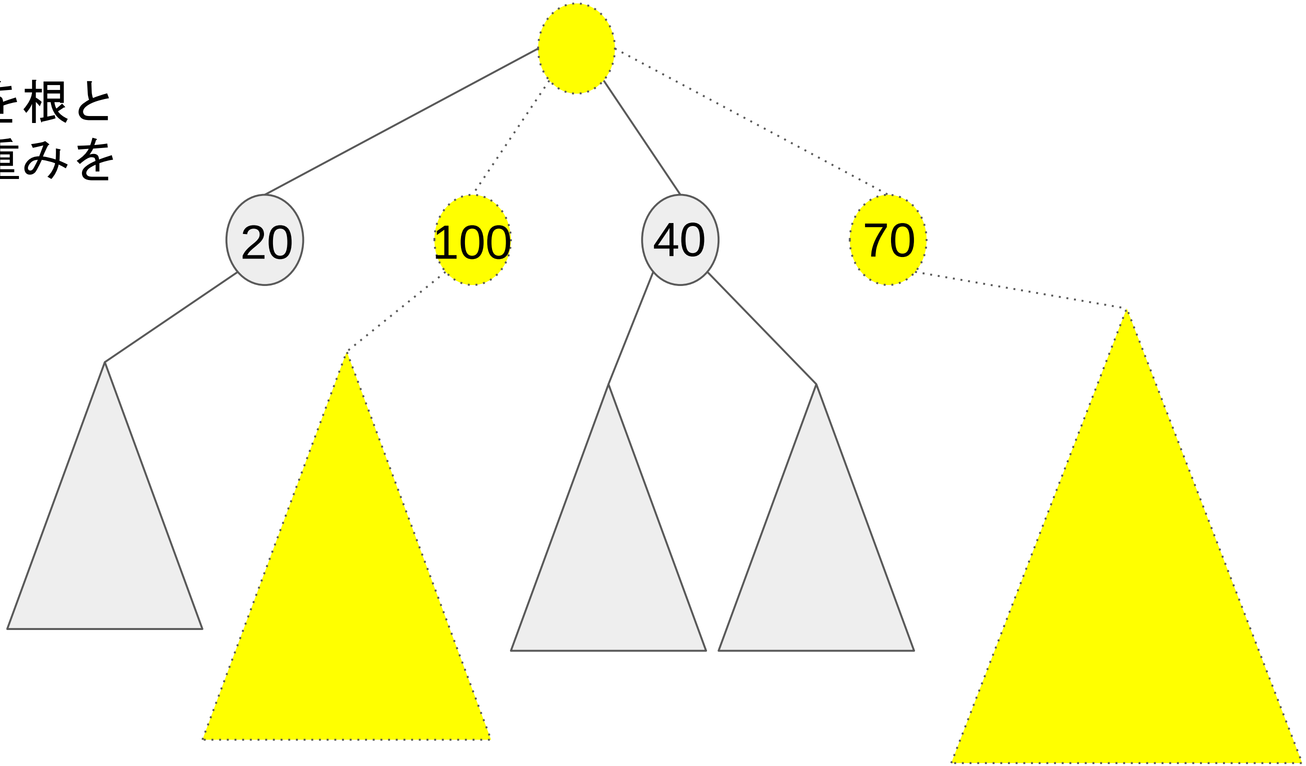
- f(v)：＝T_uの重み（uはvの兄弟）はクエリに寄らない➡︎前処理可
- 一般に、f(v)：＝T_{u_i}の重みの最大値（u_iはvの兄弟）とすれば良い
- 変更クエリがないので、この木上のRange Maximum Queryはダブリングで求められる
- **一般に、T_{x_i}の重みの最大値（x_iはパス上にないLCA(a,b)の子）を求める必要がある（超重要！！！）←上位3つまでわかってれば良い**
- max{重みの総和ーT_LCA(a,b)の重み,パス上のf(v)の最大値,
  T_{c_i}の重みの最大値（c_iはaの子）,
  T_{d_i}の重みの最大値（d_iはbの子）,
  T_{x_i}の重みの最大値（x_iはパス上にないLCA(a,b)の子）}

解法（LCA(a,b)≠aの場合）

- f(v)：＝T_uの重み（uはvの兄弟）はクエリに寄らない➡前処理可
- 一般に、f(v)：＝T_{u_i}の重みの最大値（u_iはvの兄弟）とすれば良い
- 変更クエリがないので、この木上のRange Maximum Queryはダブリングで求められる
- **一般に、T_{x_i}の重みの最大値（x_iはパス上にないLCA(a,b)の子）を求める必要がある（超重要！！！）←上位3つまでわかってれば良い**
- max{重みの総和ーT_LCA(a,b)の重み,パス上のf(v)の最大値,
  T_{c_i}の重みの最大値（c_iはaの子）,
  T_{d_i}の重みの最大値（d_iはbの子）,
  T_{x_i}の重みの最大値（x_iはパス上にないLCA(a,b)の子）}

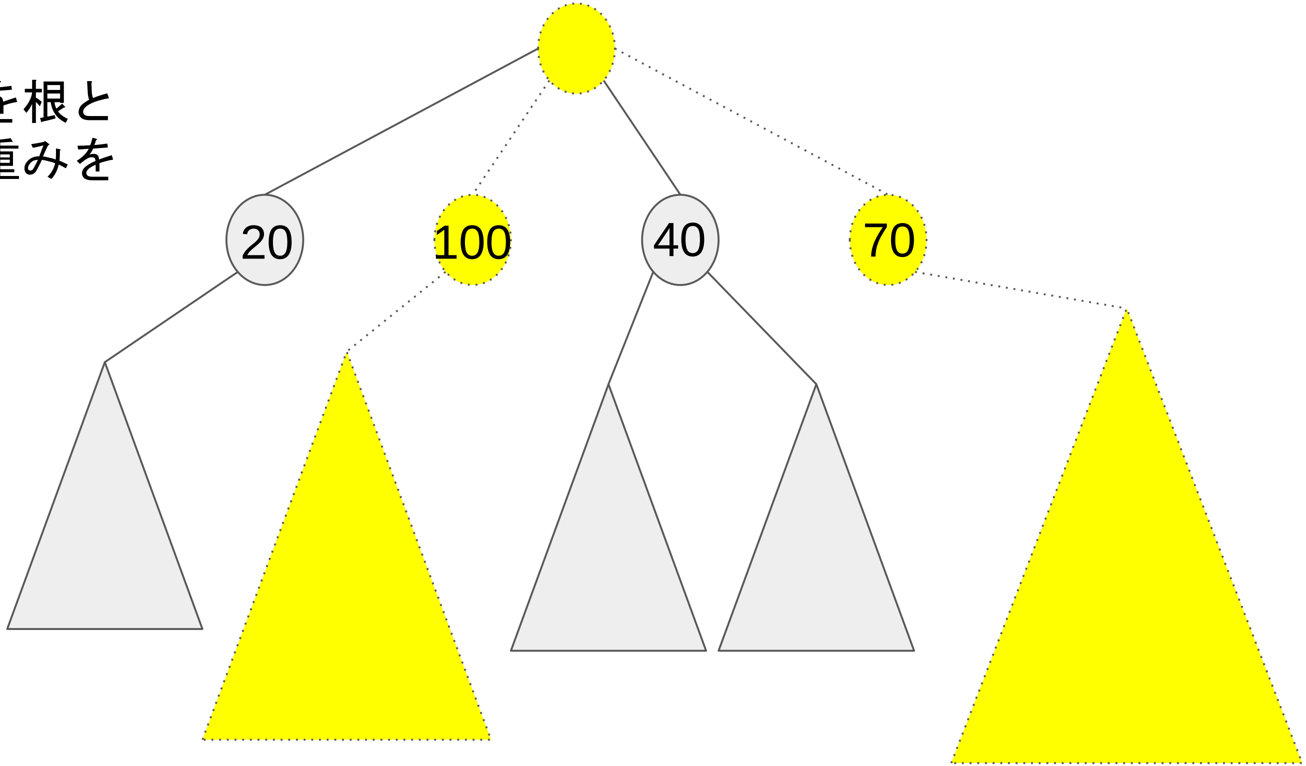
解法（LCA(a,b)≠aの場合）

※値は各頂点を根とする部分木の重みを表す



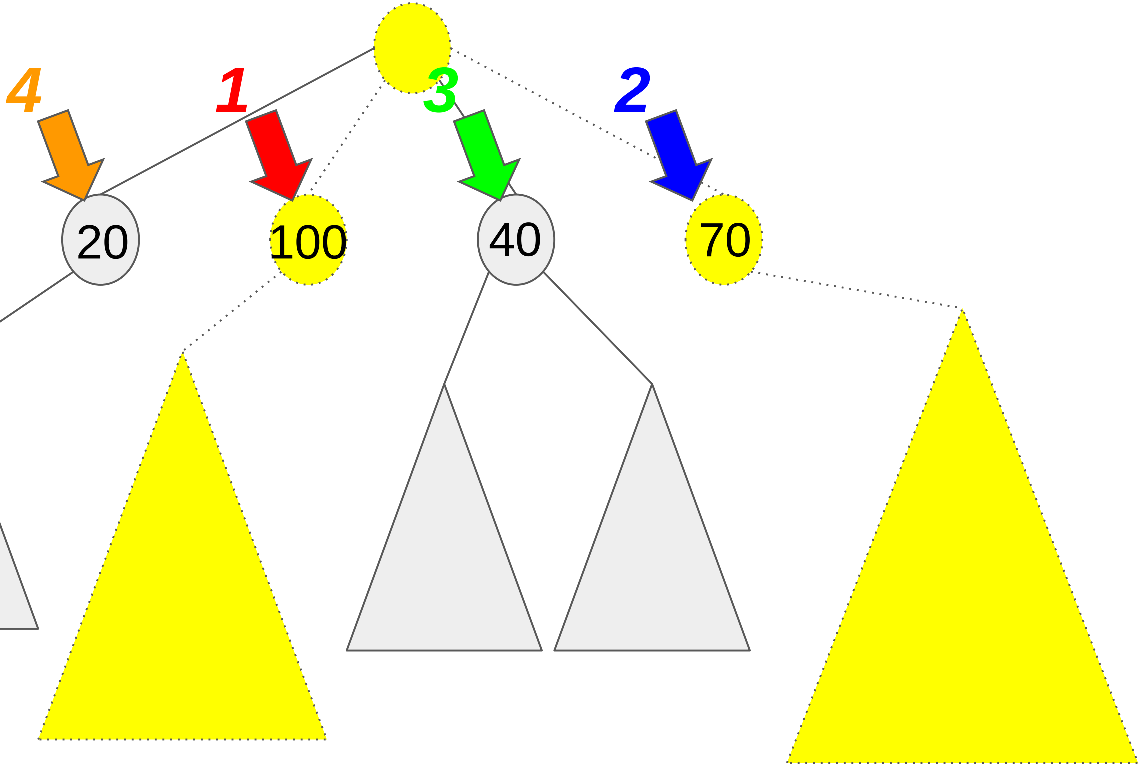
解法（LCA(a,b)≠aの場合）

※値は各頂点を根とする部分木の重みを表す



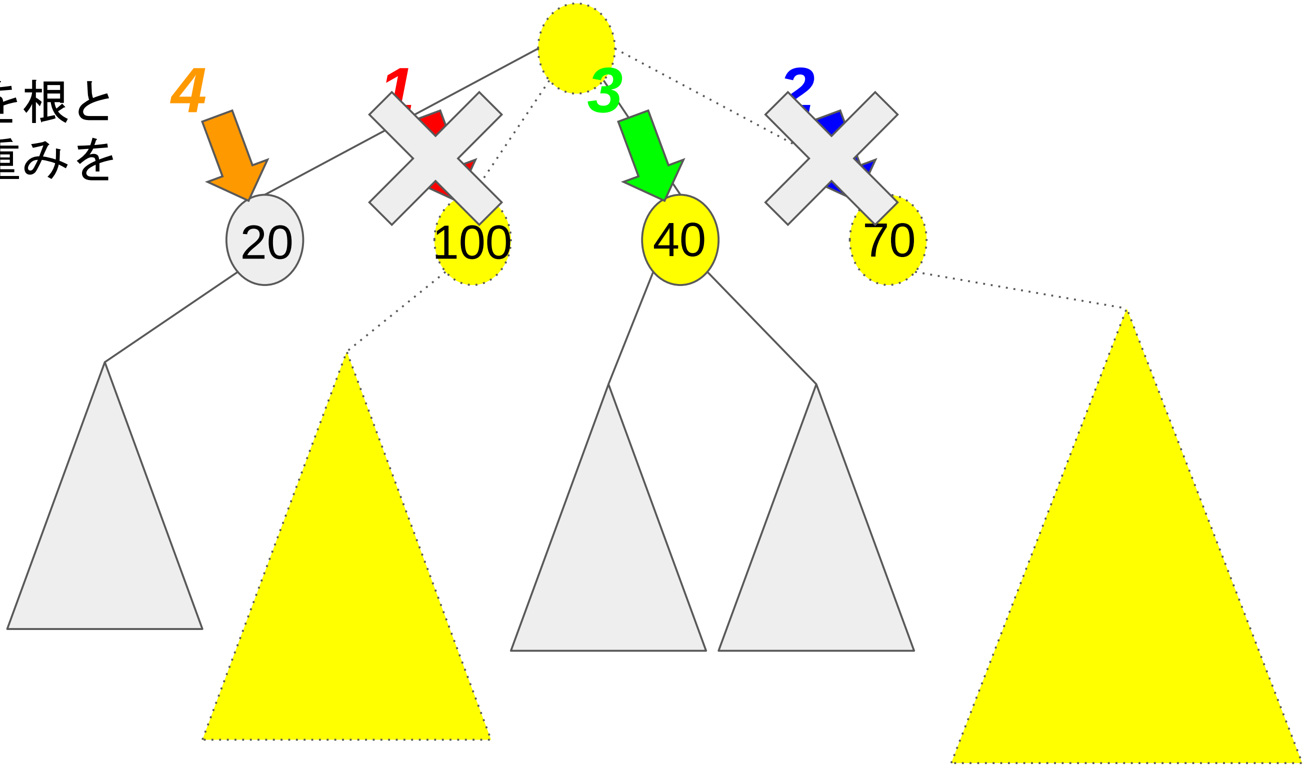
解法（LCA(a,b)≠aの場合）

※値は各頂点を根とする部分木の重みを表す



解法（LCA(a,b)≠aの場合）

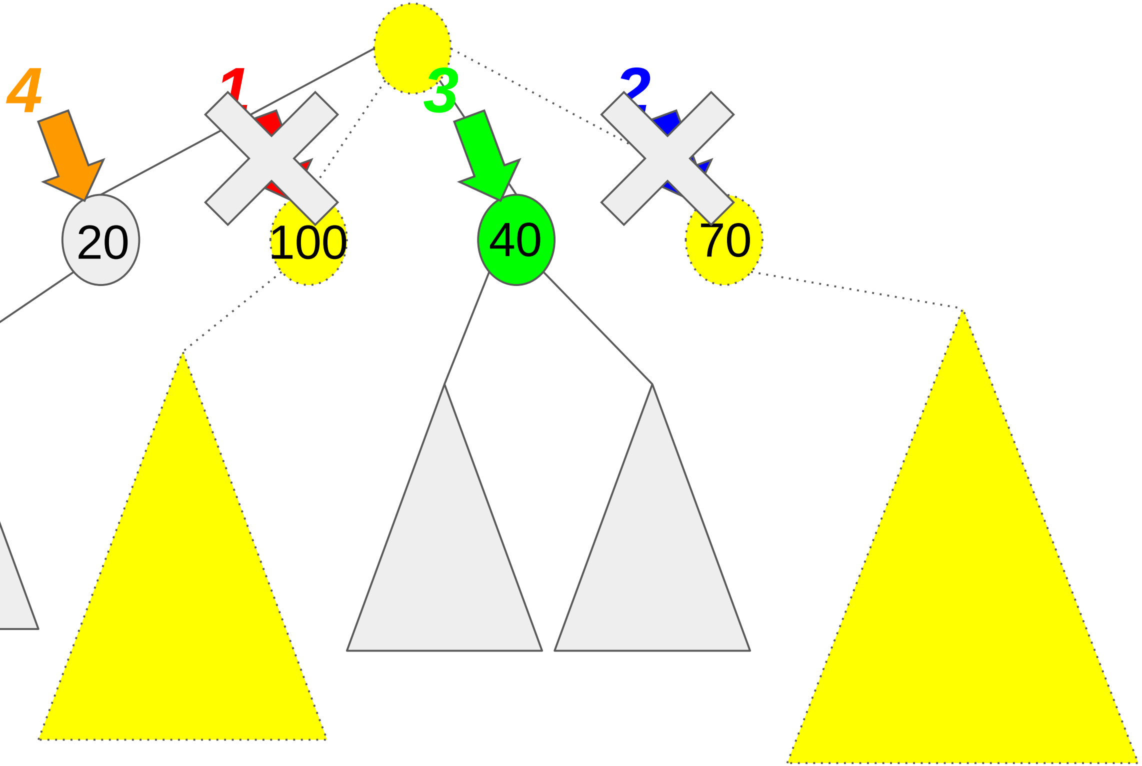
※値は各頂点を根とする部分木の重みを表す



解法（LCA(a,b)≠aの場合）

上位3つがわかれば良さそう

※値は各頂点を根とする部分木の重みを表す



終わり