データ構造とアルゴリズム
AVL 木 (平衡探索木)

岡本 吉央

豊橋技術科学大学 情報工学系

2005 年 10 月 3 日

目的

- **前回：2 分探索木：**
  最悪の場合の高さ $= O(n)$
  最良の場合の高さ $= O(\log n)$
- **今回：AVL 木**
  常に高さが $O(\log n)$ となる 2 分探索木

定義

平衡探索木 (balanced search tree) **とは**

- **根付き木**
- **高さが** $O(\log n)$

様々な平衡探索木

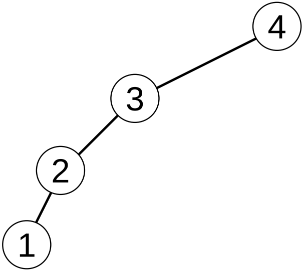
- AVL **木**
- **赤黒木** (2 **色木**)
- 2-3 **木**
- B **木**

定義

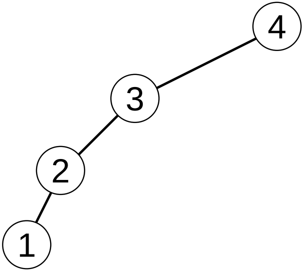
A に対する AVL 木 (AVL-tree) とは

- 2 分探索木
- どの節点においても，
  その左部分木と右部分木の高さの差が 1 以下

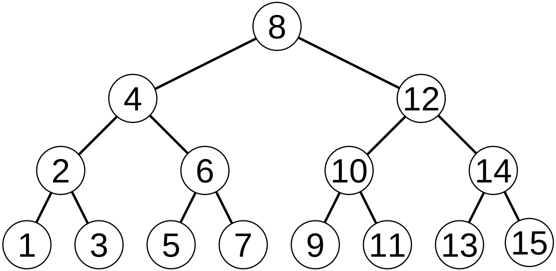
AVL 木?



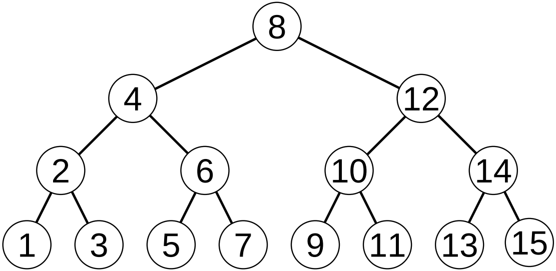
AVL 木 でない



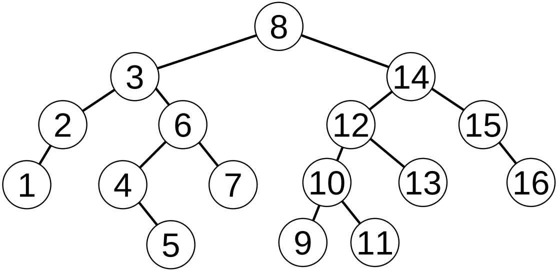
AVL 木?



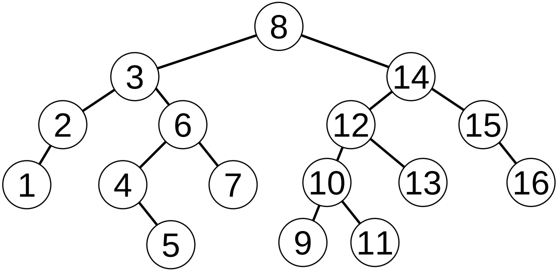
AVL 木 である



AVL 木?

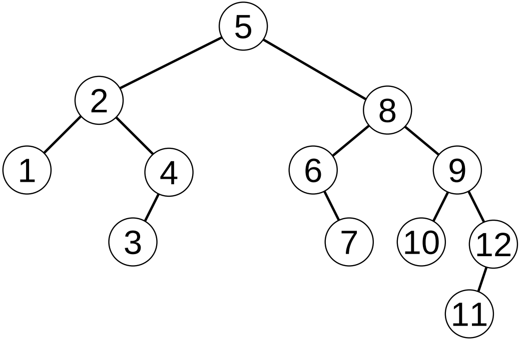


AVL 木 である

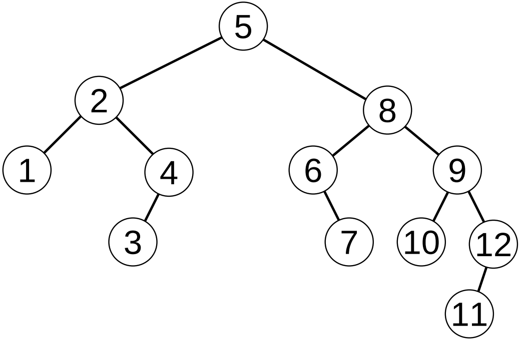


AVL 木?

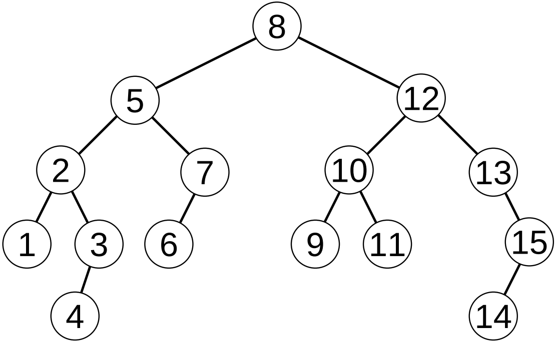
(注：配布物と異なる)



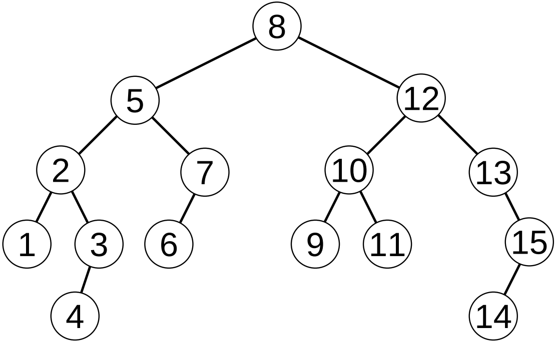
AVL 木 である　　(注：配布物と異なる)



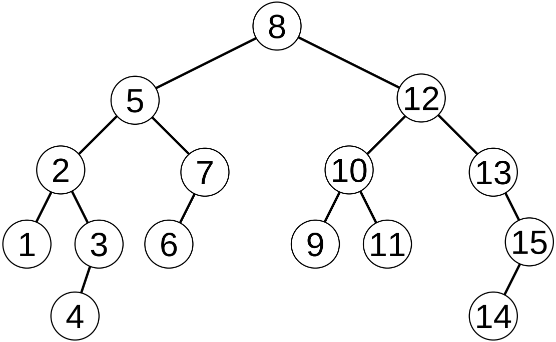
AVL 木?



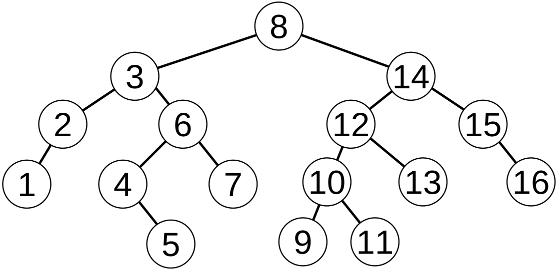
AVL 木 でない



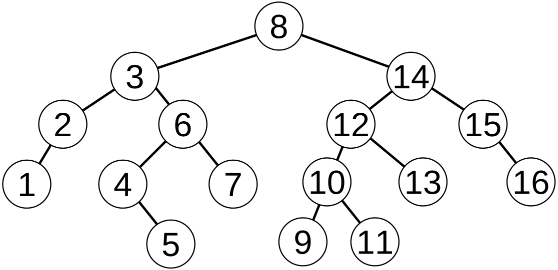
AVL 木 でない (13 に注目)



- AVL 木の部分木も AVL 木



- AVL 木の部分木も AVL 木



証明：定義から直ちに分かる

- AVL 木の高さは $O(\log n)$

- AVL 木の高さは $O(\log n)$

証明：
$f(h) =$ 高さ h の AVL 木が持つ節点の最小数

- AVL 木の高さは $O(\log n)$

証明：
$f(h) =$ 高さ h の AVL 木が持つ節点の最小数

ここで，$h \geq 2$ のとき

$$f(h) = 1 + f(h-1) + f(h-2)$$

が成り立つ (ただし，$f(0) = 1, f(1) = 2$)

解くと，

$$f(h) = \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^{h+3} - \left(\frac{1-\sqrt{5}}{2}\right)^{h+3}\right) - 1$$

解くと，

$$f(h) = \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^{h+3} - \left(\frac{1-\sqrt{5}}{2}\right)^{h+3}\right) - 1$$

ここで，$f(h) \leq n$ と置くと，

$$h = O(\log n)$$

が得られる

解くと，

$$f(h) = \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^{h+3} - \left(\frac{1-\sqrt{5}}{2}\right)^{h+3}\right) - 1$$

ここで， $f(h) \leq n$ **と置くと，**

$$h = O(\log n)$$

が得られる **[証明終]**

- MEMBER, MIN
  木を変化させないので，
  2 分探索木と同じ操作でよい

- MEMBER, MIN
  木を変化させないので，
  2 分探索木と同じ操作でよい
- INSERT，DELETE
  木を変化させるので，
  2 分探索木と同じ操作では足りない

- MEMBER, MIN
  木を変化させないので，
  2 分探索木と同じ操作でよい
- INSERT，DELETE
  木を変化させるので，
  2 分探索木と同じ操作では足りない

$\longrightarrow$ **変更操作を行なう**

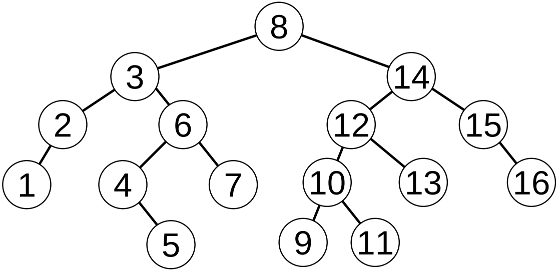
- MEMBER, MIN
  **木を変化させないので，
  2 分探索木と同じ操作でよい**
- INSERT，DELETE
  **木を変化させるので，
  2 分探索木と同じ操作では足りない**

$\longrightarrow$ **変更操作を行なう**

$\Longrightarrow$ **変更を容易にするため各節点にラベルを付ける**

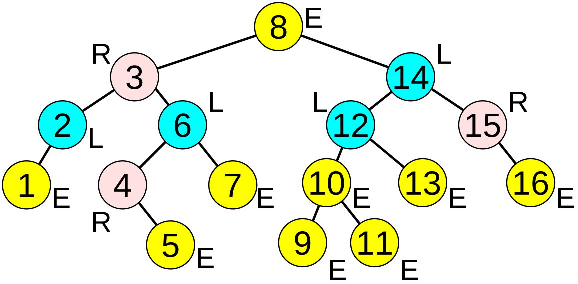
節点 y **のラベル** $s(y)$

$$s(y) = \begin{cases} L & \textbf{左部分木の高さ} > \textbf{右部分木の高さ} \\ E & \textbf{左部分木の高さ} = \textbf{右部分木の高さ} \\ R & \textbf{左部分木の高さ} < \textbf{右部分木の高さ} \end{cases}$$

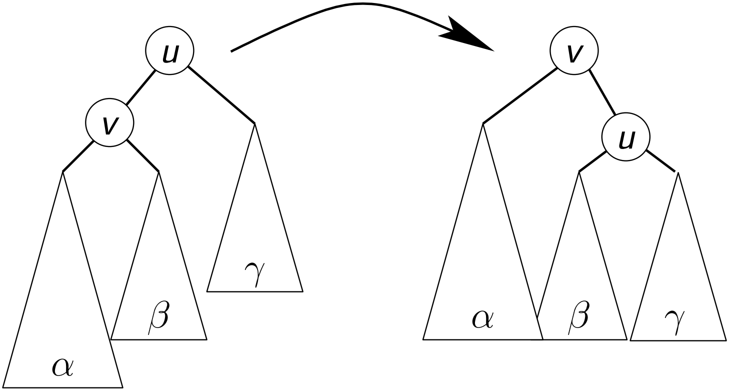


節点 y のラベル $s(y)$

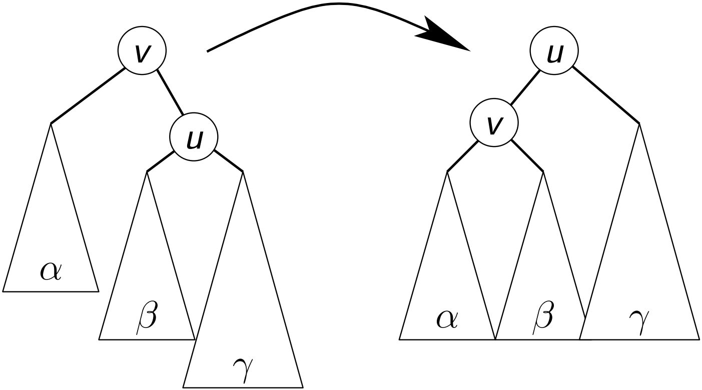
$$s(y) = \begin{cases} L & \text{左部分木の高さ} > \text{右部分木の高さ} \\ E & \text{左部分木の高さ} = \text{右部分木の高さ} \\ R & \text{左部分木の高さ} < \text{右部分木の高さ} \end{cases}$$



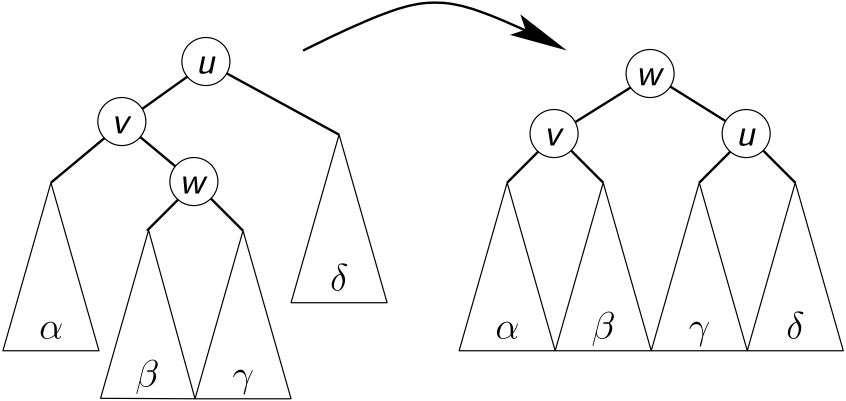
u における単右回転 (single right rotation)



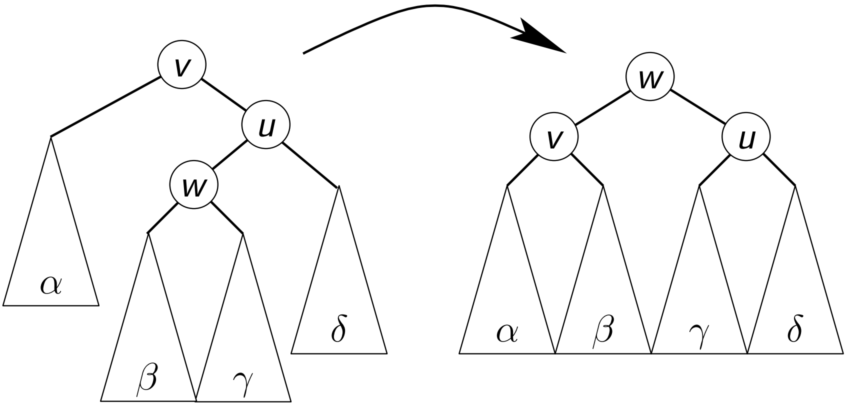
v における単左回転 (single left rotation)



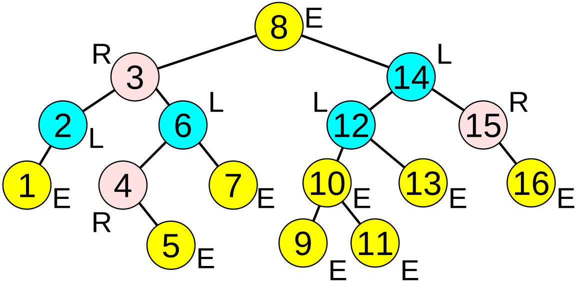
u における双左回転 (double left rotation)



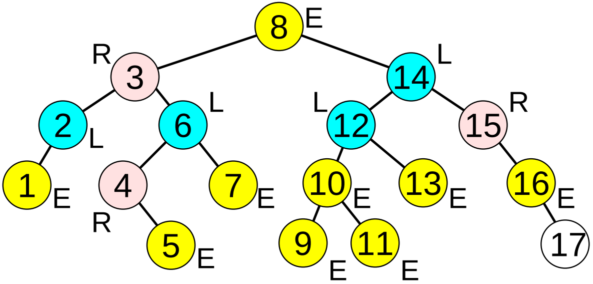
v における双右回転 (double right rotation)



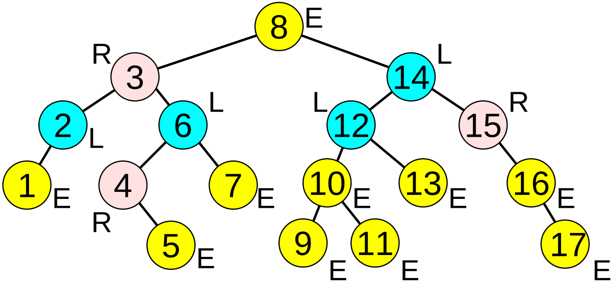
ラベルと回転の組合せによる変更：例 1



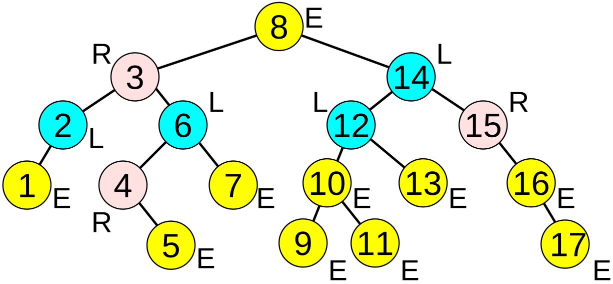
17 **の挿入**



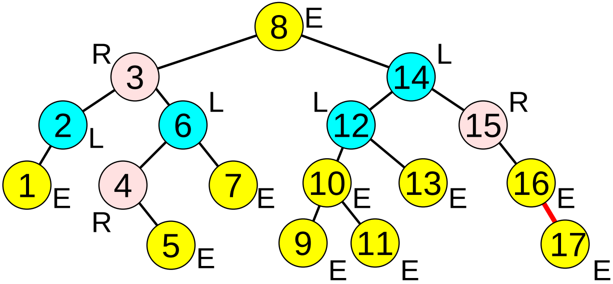
17 の挿入



17 **の挿入，** 17 **のラベルは** E (**葉だから**)



17 **から根に向かってラベルを更新**

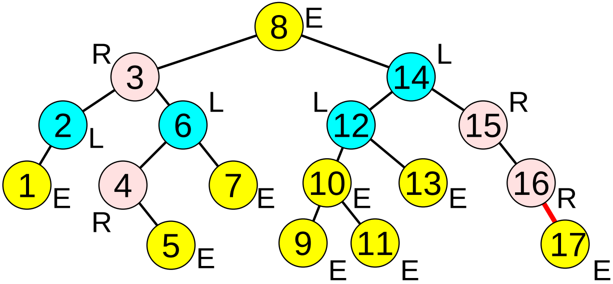


右から更新

⇒ 16 **の右部分木の高さが** 1 **増加**

16 **のラベルは** E

⇒ 16 **の新しいラベルは** R

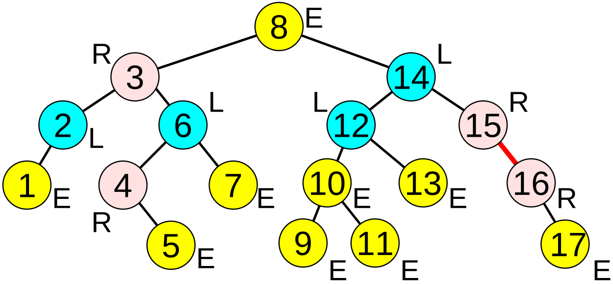


右から更新

$\Rightarrow$ 16 **の右部分木の高さが** 1 **増加**

16 **のラベルは** E

$\Rightarrow$ 16 **の新しいラベルは** R

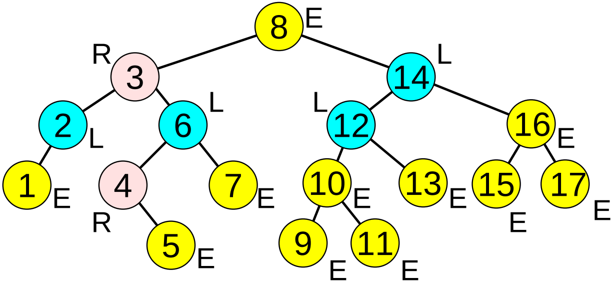


右から更新

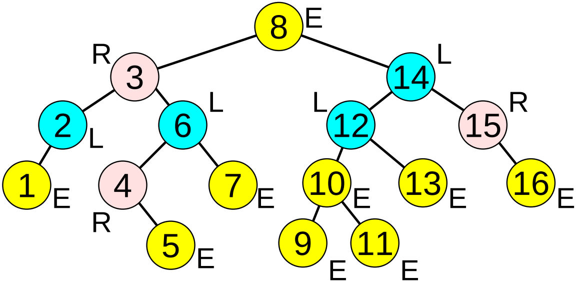
$\Rightarrow$ 15 **の右部分木の高さが** 1 **増加**

15 **のラベルは** R

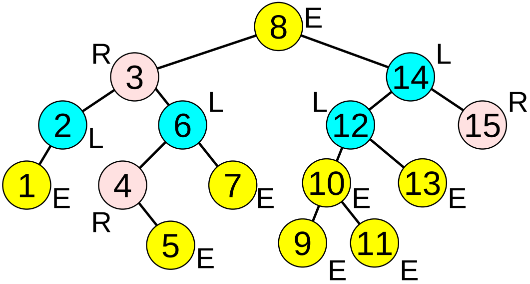
$\Rightarrow$ 15 **において条件が満たされない**



16 **のラベルは** R **だった**
$\Rightarrow$ 15 **において単左回転，ラベルも付け替え**
14 **の右部分木の高さは挿入前と変化なし**
$\Rightarrow$ **変更終了**

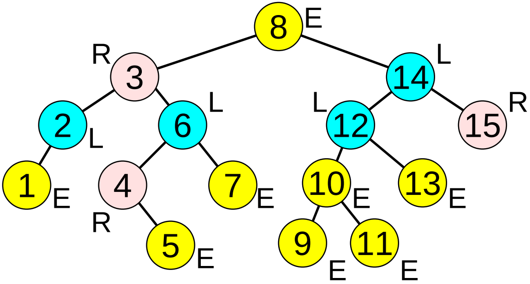


16 の削除

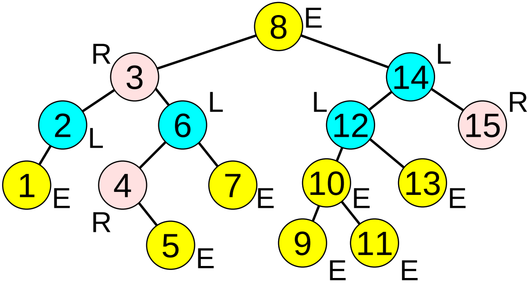


16 の削除

ラベルと回転の組合せによる変更：例 2



16 **の親だった** 15 **からラベルの付け替え**

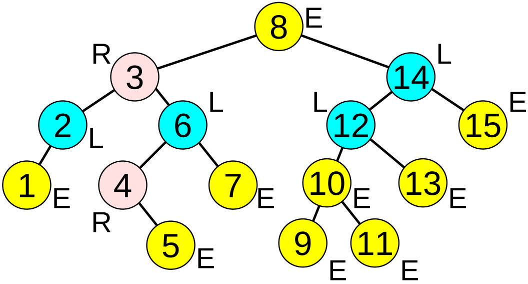


右から更新

$\Rightarrow$ 15 **の右部分木の高さが** 1 **減少**

15 **のラベルは** R

$\Rightarrow$ 15 **の新しいラベルは** E

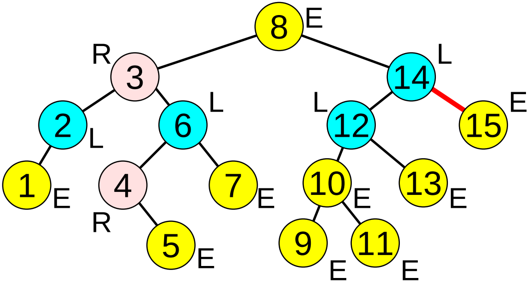


右から更新

⇒ 15 **の右部分木の高さが** 1 **減少**

15 **のラベルは** R

⇒ 15 **の新しいラベルは** E



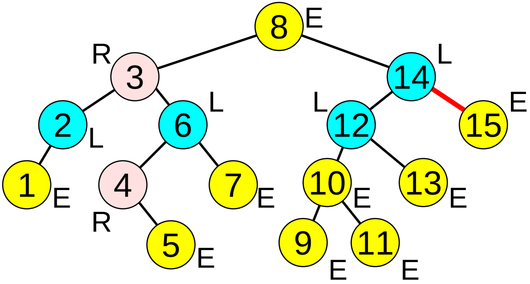
右から更新

$\Rightarrow$ 14 **の右部分木の高さが** 1 **減少**

14 **のラベルは** L

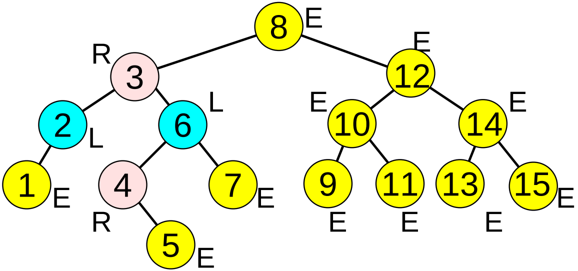
$\Rightarrow$ 14 **において条件が満たされない**

ラベルと回転の組合せによる変更：例 2

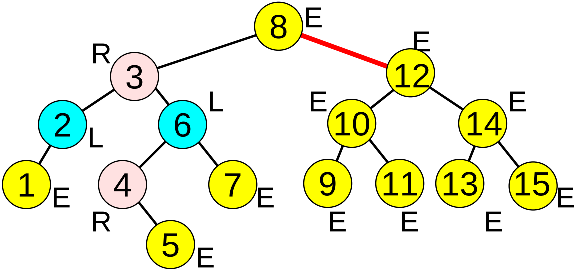


15 **のラベルは** E **だった**

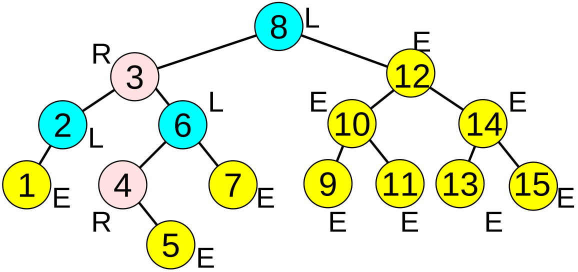
$\Rightarrow$ 14 **において単右回転，ラベルも付け替え**



15 **のラベルは** E **だった**
$\Rightarrow$ 14 **において単右回転，ラベルも付け替え**



右から更新
⇒ **右部分木の高さが** 1 **減少**
8 **のラベルは** E
⇒ 8 **の新しいラベルは** L

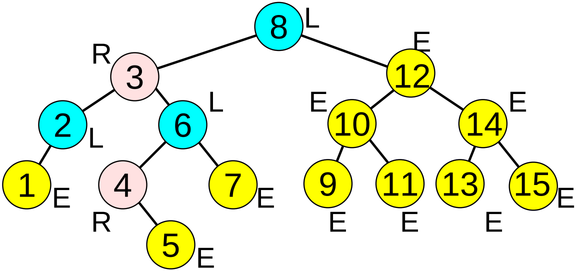


右から更新

⇒ **右部分木の高さが** 1 **減少**

8 **のラベルは** E

⇒ 8 **の新しいラベルは** L



8 **を根とする部分木の高さは変化なし**
$\Rightarrow$ **全体の変更を終了**

一般的な挿入と削除のアルゴリズム (概要)

- 要素の挿入，削除
- $\Rightarrow$ 部分木の高さが変化
- $\Rightarrow$ ラベルを変更，必要ならば回転操作

一般的な挿入と削除のアルゴリズム (概要)

- 要素の挿入，削除
- $\Rightarrow$ 部分木の高さが変化
- $\Rightarrow$ ラベルを変更，必要ならば回転操作

注意

- アルゴリズムは暗記しない
- 論理的に考えれば導くことが出来る

挿入した要素のラベルは E
そこから根に向かって順に昇る

1. **ある節点 v に右から到達したとき (すなわち，右部分木の高さが 1 増えたとき)**

挿入した要素のラベルは E
そこから根に向かって順に昇る

1. ある節点 v に右から到達したとき
(すなわち，右部分木の高さが 1 増えたとき)

1-1. v のラベルが L のとき
v を根とする部分木の高さは不変 $\Rightarrow$
v のラベルを E に変更して，変更終了

挿入した要素のラベルは E
そこから根に向かって順に昇る

1. **ある節点 v に右から到達したとき**
   (すなわち，右部分木の高さが 1 増えたとき)

1-1. v **のラベルが L のとき**
   v **を根とする部分木の高さは不変** $\Rightarrow$
   v **のラベルを E に変更して，変更終了**

1-2. v **のラベルが E のとき**
   v **を根とする部分木の高さが変化** $\Rightarrow$
   v **のラベルを R にして，上へ昇る**

挿入した要素のラベルは E
そこから根に向かって順に昇る

1. **ある節点 v に右から到達したとき**
   (すなわち，右部分木の高さが 1 増えたとき)

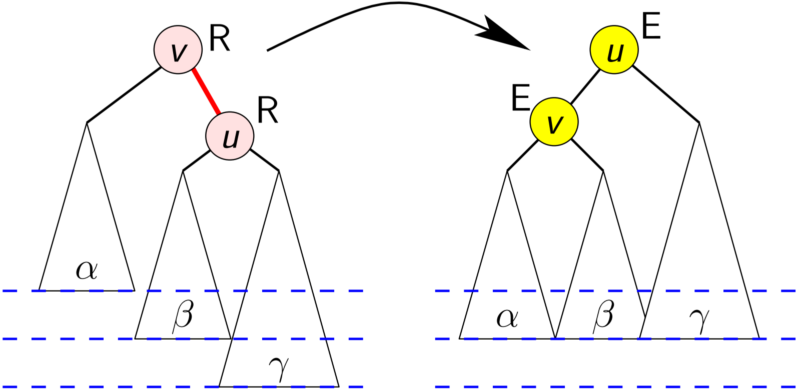
1-1. v **のラベルが L のとき**
   v **を根とする部分木の高さは不変** $\Rightarrow$
   v **のラベルを E に変更して，変更終了**

1-2. v **のラベルが E のとき**
   v **を根とする部分木の高さが変化** $\Rightarrow$
   v **のラベルを R にして，上へ昇る**

1-3. v **のラベルが R のとき**
   v **で条件が満たされない** $\longrightarrow$

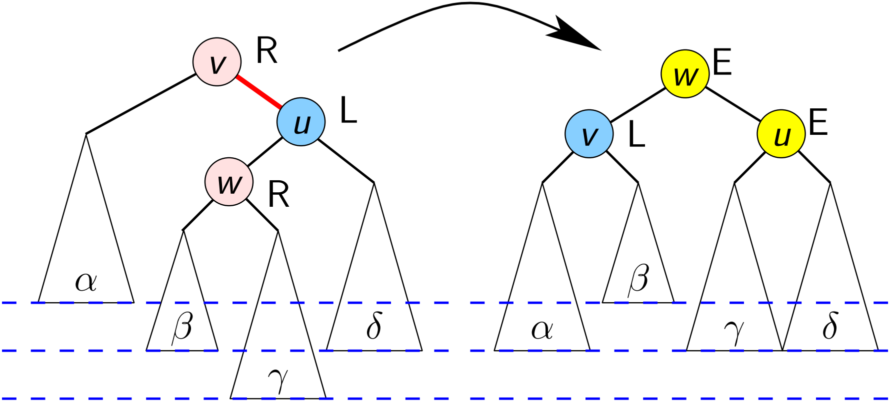
1-3. v のラベルが R のとき

1-3-1. v の右の子供 u のラベルが R のとき
v において単左回転，ラベルも付け替え
変更終了



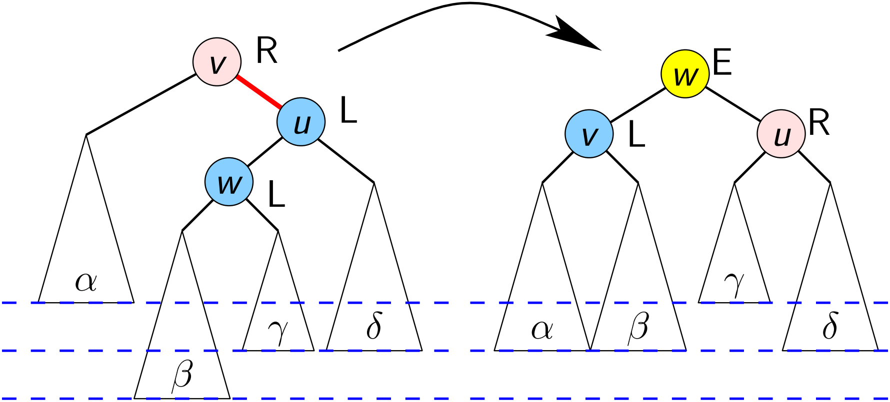
1-3. v のラベルが R のとき

1-3-2. v の右の子供 u のラベルが L のとき
v において双右回転，ラベルも付け替え
変更終了



1-3. v のラベルが R のとき

1-3-2. v の右の子供 u のラベルが L のとき
v において双右回転，ラベルも付け替え
変更終了



挿入した要素のラベルは E
そこから根に向かって順に昇る

2. **ある節点 v に左から到達したとき**
(すなわち，左部分木の高さが 1 増えたとき)

挿入した要素のラベルは E
そこから根に向かって順に昇る

2. **ある節点 v に左から到達したとき**
   (すなわち，左部分木の高さが 1 増えたとき)

2-1. **v のラベルが R のとき**
   v を根とする部分木の高さは不変 $\Rightarrow$
   v のラベルを E に変更して，変更終了

挿入した要素のラベルは E
そこから根に向かって順に昇る

2. **ある節点 v に左から到達したとき**
   (すなわち，左部分木の高さが 1 増えたとき)

2-1. v **のラベルが R のとき**
   v **を根とする部分木の高さは不変** $\Rightarrow$
   v **のラベルを E に変更して，変更終了**

2-2. v **のラベルが E のとき**
   v **を根とする部分木の高さが変化** $\Rightarrow$
   v **のラベルを L にして，上へ昇る**

挿入した要素のラベルは E
そこから根に向かって順に昇る

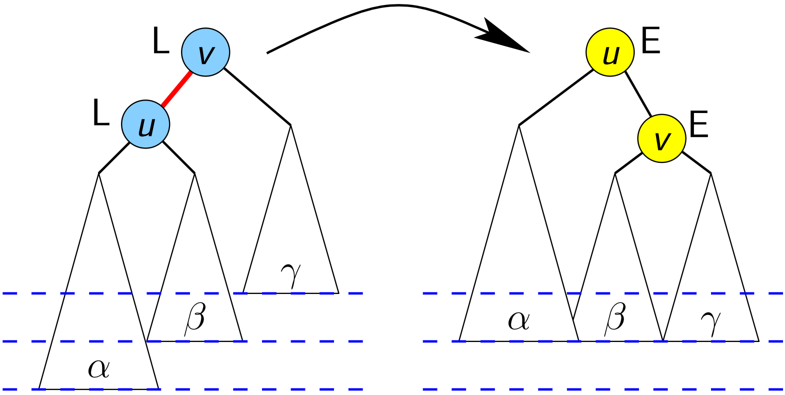
2. **ある節点 v に左から到達したとき**
   (すなわち，左部分木の高さが 1 増えたとき)

2-1. v **のラベルが R のとき**
   v **を根とする部分木の高さは不変** $\Rightarrow$
   v **のラベルを E に変更して，変更終了**

2-2. v **のラベルが E のとき**
   v **を根とする部分木の高さが変化** $\Rightarrow$
   v **のラベルを L にして，上へ昇る**

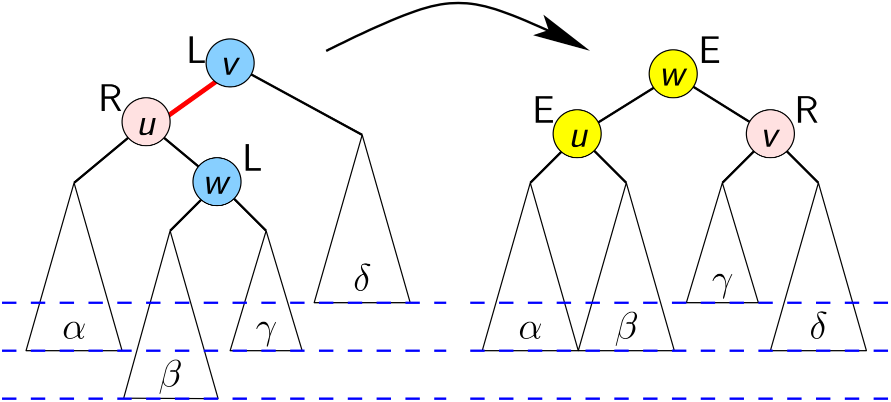
2-3. v **のラベルが L のとき**
   v **で条件が満たされない** $\longrightarrow$

2-3. v のラベルが L のとき
2-3-1. v の左の子供 u のラベルが L のとき
v において単右回転，ラベルも付け替え
変更終了



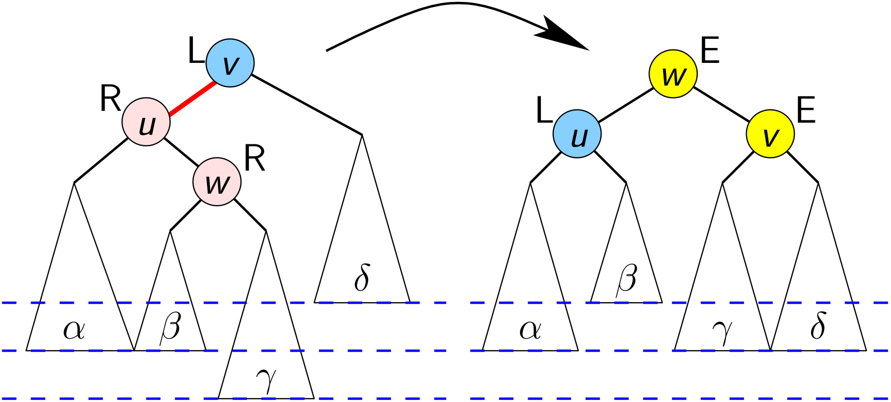
2-3. v **のラベルが** L **のとき**

2-3-2. v **の左の子供** u **のラベルが** R **のとき**
v **において双左回転，ラベルも付け替え**
変更終了



2-3. v のラベルが L のとき

2-3-2. v の右の子供 u のラベルが R のとき
v において双左回転，ラベルも付け替え
変更終了



削除要素を移動させた場所の親から根に向かって順に昇る

1. **ある節点 v に左から到達したとき（すなわち，左部分木の高さが 1 減ったとき）**

削除要素を移動させた場所の親から根に向かって順に昇る

1. **ある節点 v に左から到達したとき（すなわち，左部分木の高さが 1 減ったとき）**

1-1. **v のラベルが L のとき**
v のラベルを E に変更して，上へ昇る

削除要素を移動させた場所の親から根に向かって順に昇る

1. **ある節点 v に左から到達したとき**
   (すなわち，左部分木の高さが 1 減ったとき)

1-1. **v のラベルが L のとき**
v のラベルを E に変更して，上へ昇る

1-2. **v のラベルが E のとき**
v のラベルを R にして，変更終了

一般的な変更の方法：削除

削除要素を移動させた場所の親から根に向かって順に昇る

1. **ある節点 v に左から到達したとき**
   (すなわち，左部分木の高さが 1 減ったとき)

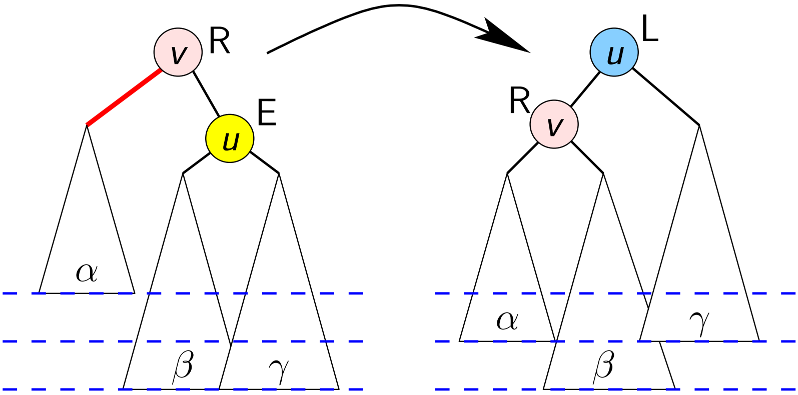
1-1. **v のラベルが L のとき**
   v のラベルを E に変更して，上へ昇る

1-2. **v のラベルが E のとき**
   v のラベルを R にして，変更終了

1-3. **v のラベルが R のとき**
   v で条件が満たされない $\longrightarrow$

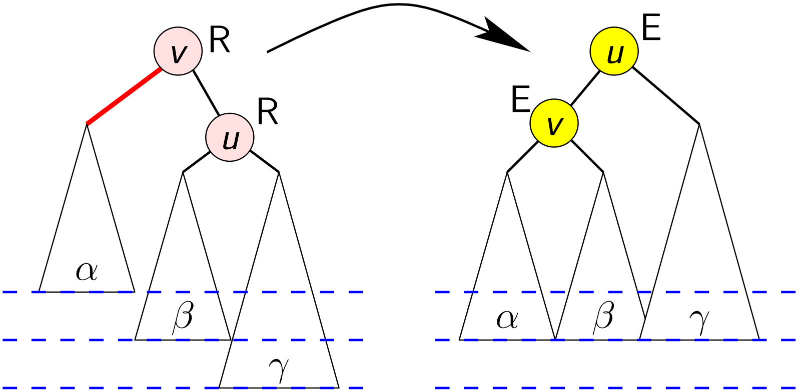
1-3. v のラベルが R のとき

1-3-1. v の右の子供 u のラベルが E のとき
v において単左回転，ラベルも付け替え
変更終了



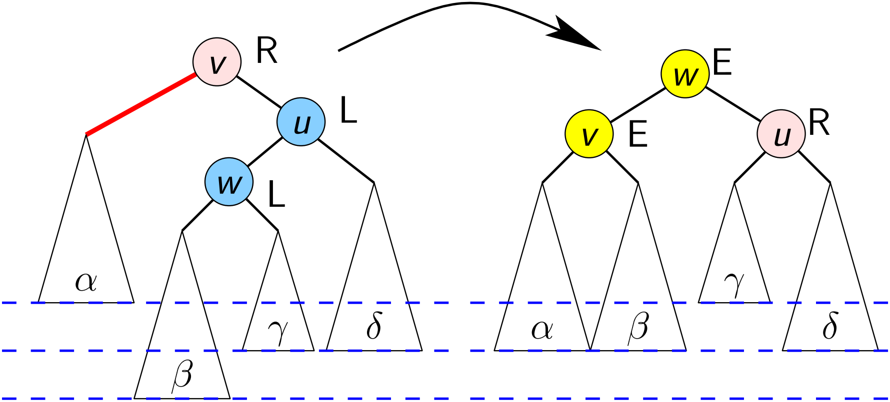
1-3. v のラベルが R のとき

1-3-2. v の右の子供 u のラベルが R のとき
v において単左回転，ラベルも付け替え
上へ昇る



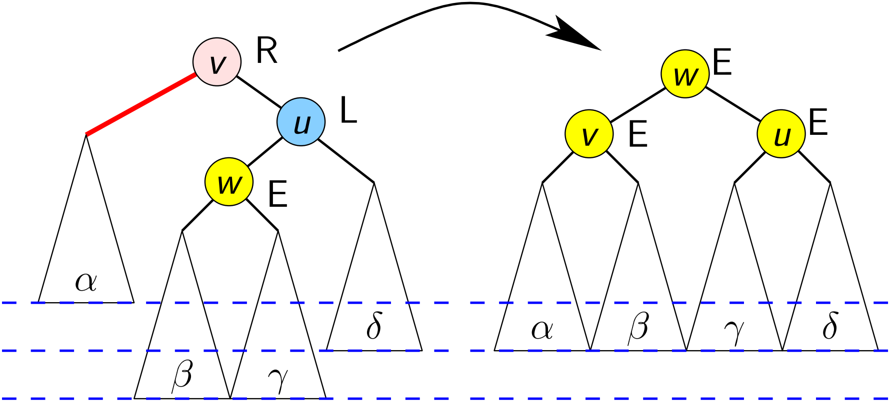
1-3. v のラベルが R のとき

1-3-3. v の右の子供 u のラベルが L のとき
v において双右回転，ラベルも付け替え
上へ昇る



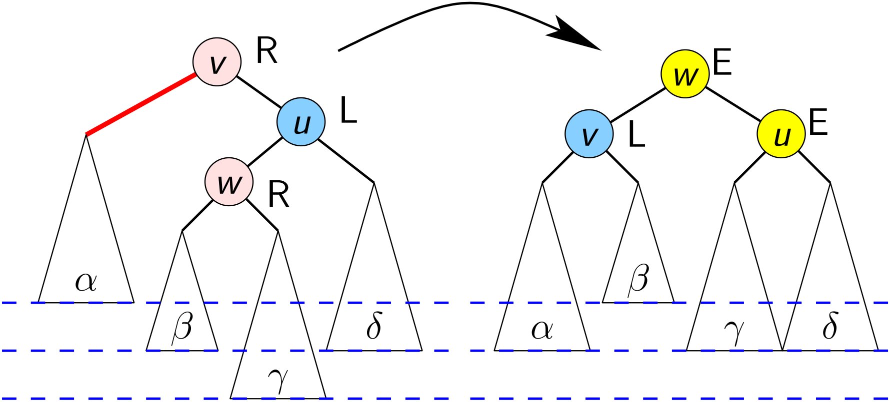
1-3. v のラベルが R のとき

1-3-3. v の右の子供 u のラベルが L のとき
v において双右回転，ラベルも付け替え
上へ昇る



1-3. v のラベルが R のとき

1-3-3. v の右の子供 u のラベルが L のとき
v において双右回転，ラベルも付け替え
上へ昇る



削除要素を移動させた場所の親から根に向かって順に昇る

2. **ある節点 v に右から到達したとき（すなわち，右部分木の高さが 1 減ったとき）**

削除要素を移動させた場所の親から根に向かって順に昇る

2. **ある節点 v に右から到達したとき**
   (すなわち，右部分木の高さが 1 減ったとき)

2-1. **v のラベルが R のとき**
   v のラベルを E に変更して，上へ昇る

一般的な変更の方法：削除

削除要素を移動させた場所の親から
根に向かって順に昇る

2. **ある節点 v に右から到達したとき**
   (すなわち，右部分木の高さが 1 減ったとき)

2-1. v **のラベルが** R **のとき**
   v **のラベルを** E **に変更して，上へ昇る**

2-2. v **のラベルが** E **のとき**
   v **のラベルを** L **にして，変更終了**

削除要素を移動させた場所の親から
根に向かって順に昇る

2. ある節点 v に右から到達したとき
(すなわち，右部分木の高さが 1 減ったとき)

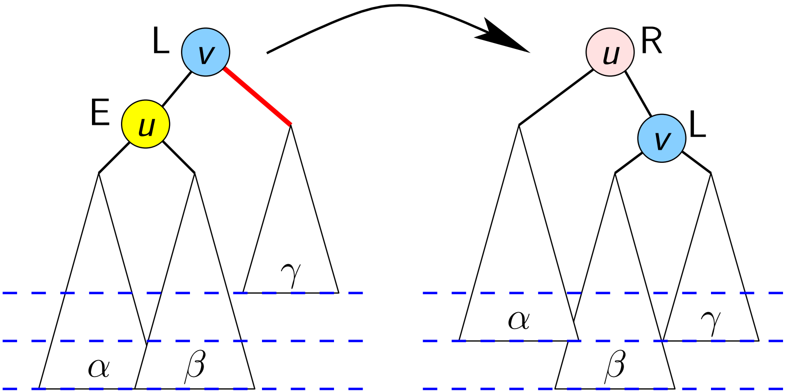
2-1. v のラベルが R のとき
v のラベルを E に変更して，上へ昇る

2-2. v のラベルが E のとき
v のラベルを L にして，変更終了

2-3. v のラベルが L のとき
v で条件が満たされない $\longrightarrow$

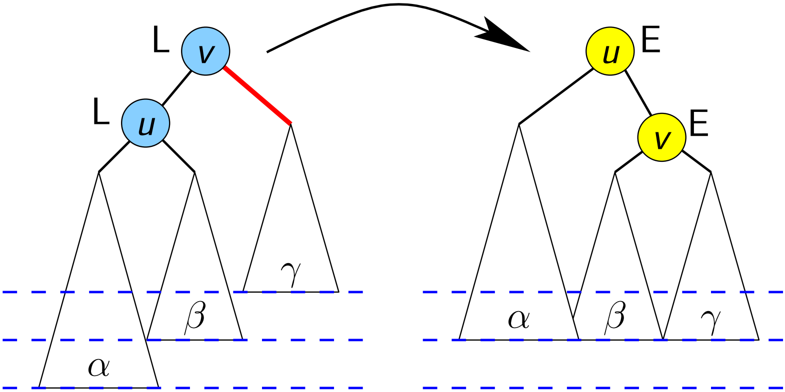
2-3. v のラベルが L のとき

2-3-1. v の左の子供 u のラベルが E のとき
v において単右回転，ラベルも付け替え
変更終了



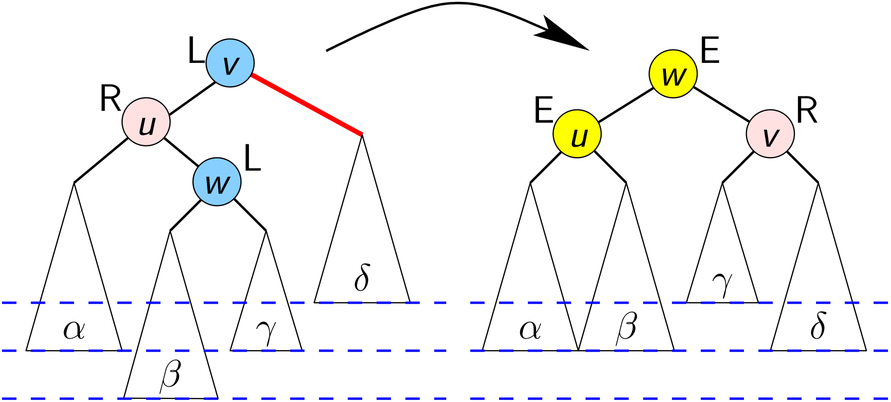
2-3. v のラベルが L のとき

2-3-2. v の左の子供 u のラベルが L のとき
v において単右回転，ラベルも付け替え
上へ昇る



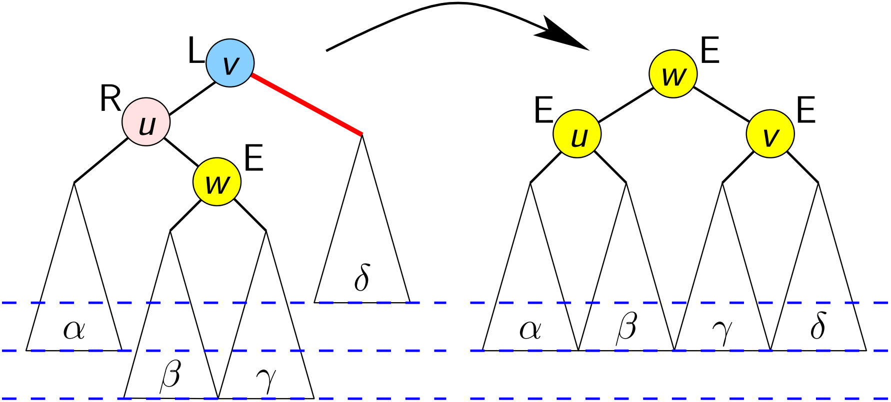
2-3. v のラベルが L のとき

2-3-3. v の左の子供 u のラベルが R のとき
v において双左回転，ラベルも付け替え
上へ昇る



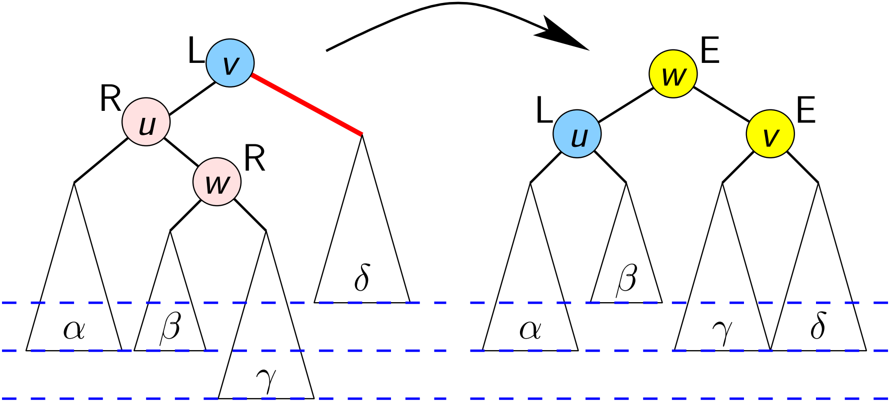
2-3. v のラベルが L のとき

2-3-3. v の左の子供 u のラベルが R のとき
v において双左回転，ラベルも付け替え
上へ昇る



2-3. v のラベルが L のとき

2-3-3. v の左の子供 u のラベルが R のとき
v において双左回転，ラベルも付け替え
上へ昇る



- MEMBER，MIN
  2 分探索木と同じく，$O(\text{木の高さ}) = O(\log n)$

- MEMBER，MIN
  2 分探索木と同じく，$O(\text{木の高さ}) = O(\log n)$
- INSERT，DELETE
  挿入と削除自体に，$O(\text{木の高さ}) = O(\log n)$
  各回転は定数個のポインタの付け替え
  回転操作の数は高々 $O(\text{木の高さ})$
  $\therefore$ 計算量は $O(\text{木の高さ}) = O(\log n)$

まとめ

- **平衡探索木と** AVL **木**

まとめ

- **平衡探索木と** AVL **木**
- AVL **木の高さ** $= \Theta(\log n)$

まとめ

- 平衡探索木と AVL 木
- AVL 木の高さ $= \Theta(\log n)$
- 回転操作

まとめ

- **平衡探索木と** AVL **木**
- AVL **木の高さ** $= \Theta(\log n)$
- **回転操作**
- **ラベルによる操作の効率化**

まとめ

- 平衡探索木と AVL 木
- AVL 木の高さ $= \Theta(\log n)$
- 回転操作
- ラベルによる操作の効率化
- (操作は覚えるのではなく，論理的に導く)

演習問題 (1)

$\{1, 2, 3, 4\}$ に対する 2 分探索木を全て書き下してみよ．

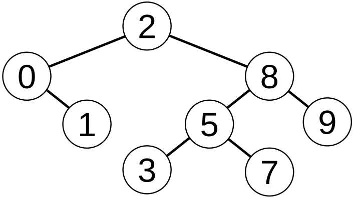
- 2 分探索木は全部でいくつあるだろうか?
- その中で AVL 木であるものはどれか?
- AVL 木はいくつあるだろうか?
- ランダム 2 分探索木のように，ランダムな順列に従って要素を挿入することによって AVL 木を構成した場合，各 AVL 木はどれほどの確率で得られるだろうか?

2 分探索木に対する各回転操作が「2 分探索木である」という性質を保つことを示せ

演習問題 (3)

挿入操作における 1-3 と 2-3 の場合において，節点 w のラベルが E である場合を考える必要がないのはなぜか?

下の AVL **木に対して** (A)〜(F) **の操作を順に行なうと，各操作後に得られる** AVL **木はどうなるか**?



(A)	6 **の挿入**	(B)	1 **の削除**	(C)	4 **の挿入**
(D)	3 **の削除**	(E)	8 **の削除**	(F)	1 **の挿入**

異なる整数の集合 A **と** B**，そして，整数** x **に対して，次の不等式が成り立っているとする．**

- **任意の** $a \in A$ **と** $b \in B$ **に対して，** $a < x < b$**．**

A **に対する** AVL **木と** B **に対する** AVL **木が与えられていて，それぞれの高さは** h_A, h_B **とする．**
このとき， $A \cup \{x\} \cup B$ **に対する** AVL **木を**

- $O(1 + |h_A - h_B|)$ **という計算量**

で構成するにはどうしたらよいか?
(h_A **と** h_B **は事前に分かっているとする．**)