

データ構造とアルゴリズム

7. 平衡木 (AVL木)

The background is a deep blue gradient. It features several thin, curved, light blue streaks that sweep across the frame. In the lower right quadrant, there is a bright, glowing circular shape composed of multiple overlapping, wavy lines, creating a sense of depth and light. Scattered throughout the background are numerous small, white, star-like points of light.

<http://numata.designed.jp/dg/>

平衡木



平衡木 (Balanced Tree) 4

- 挿入・削除を行う度に木の形を整えて、常に探索が $O(\lg n)$ で行えるようにする木。
- 整形に探索以上のコストをかけては元も子もないので、整形のコストは $O(\lg n)$ 以下でなければいけない。

平衡木の種類

- **AVL木**
- **B木**
- **2-3木**
- **2-3-4木**
- **赤黒木**
- **スプレー木**

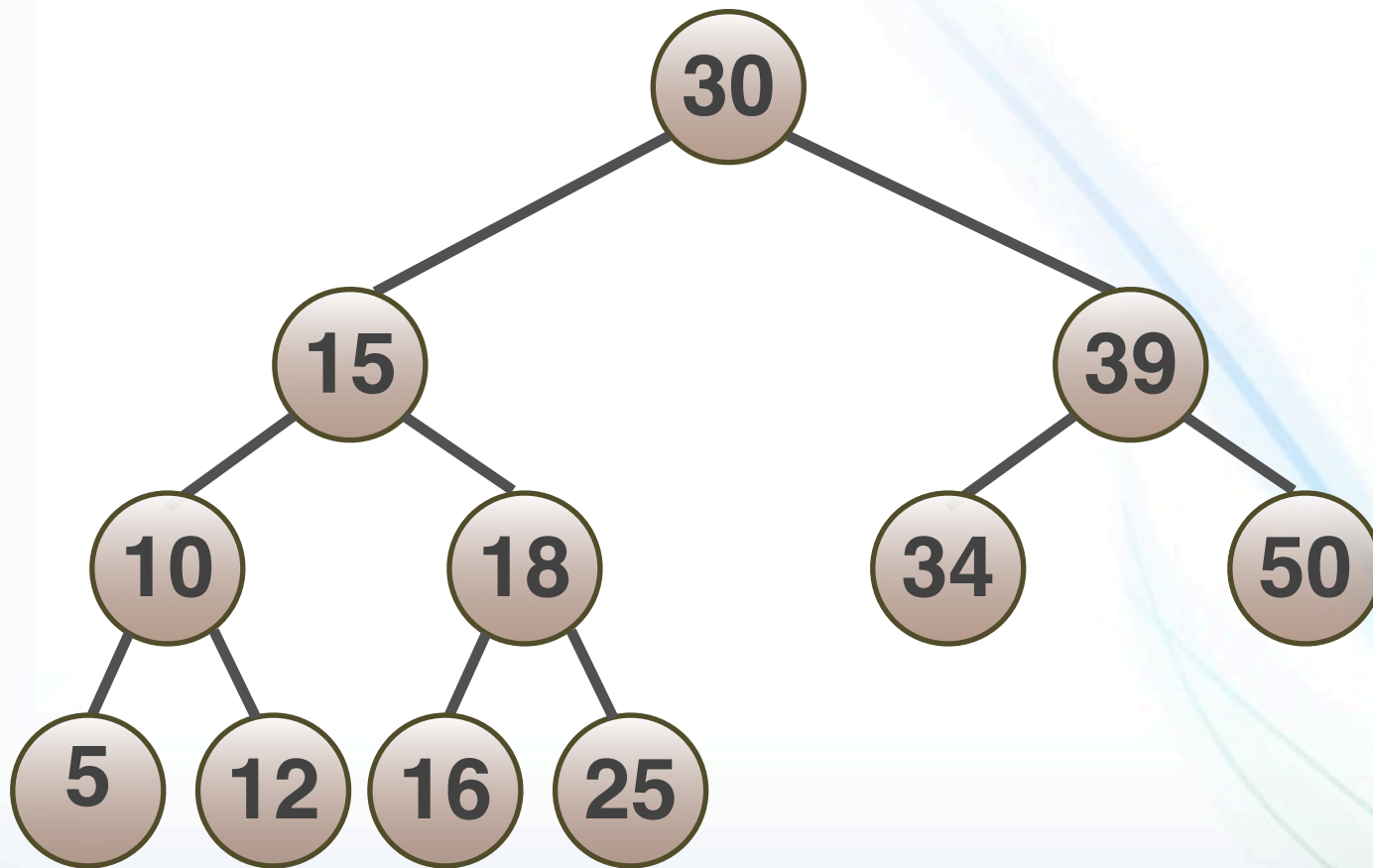
AVL木

The background is a deep blue gradient. It features several glowing, ethereal light trails in shades of cyan and white. One prominent trail forms a large, swirling, circular shape on the right side of the image. Other smaller, more linear trails are scattered across the left and bottom portions. Numerous small, bright white dots, resembling stars or distant galaxies, are sparsely distributed throughout the blue field.

予備知識：部分木の表現

7

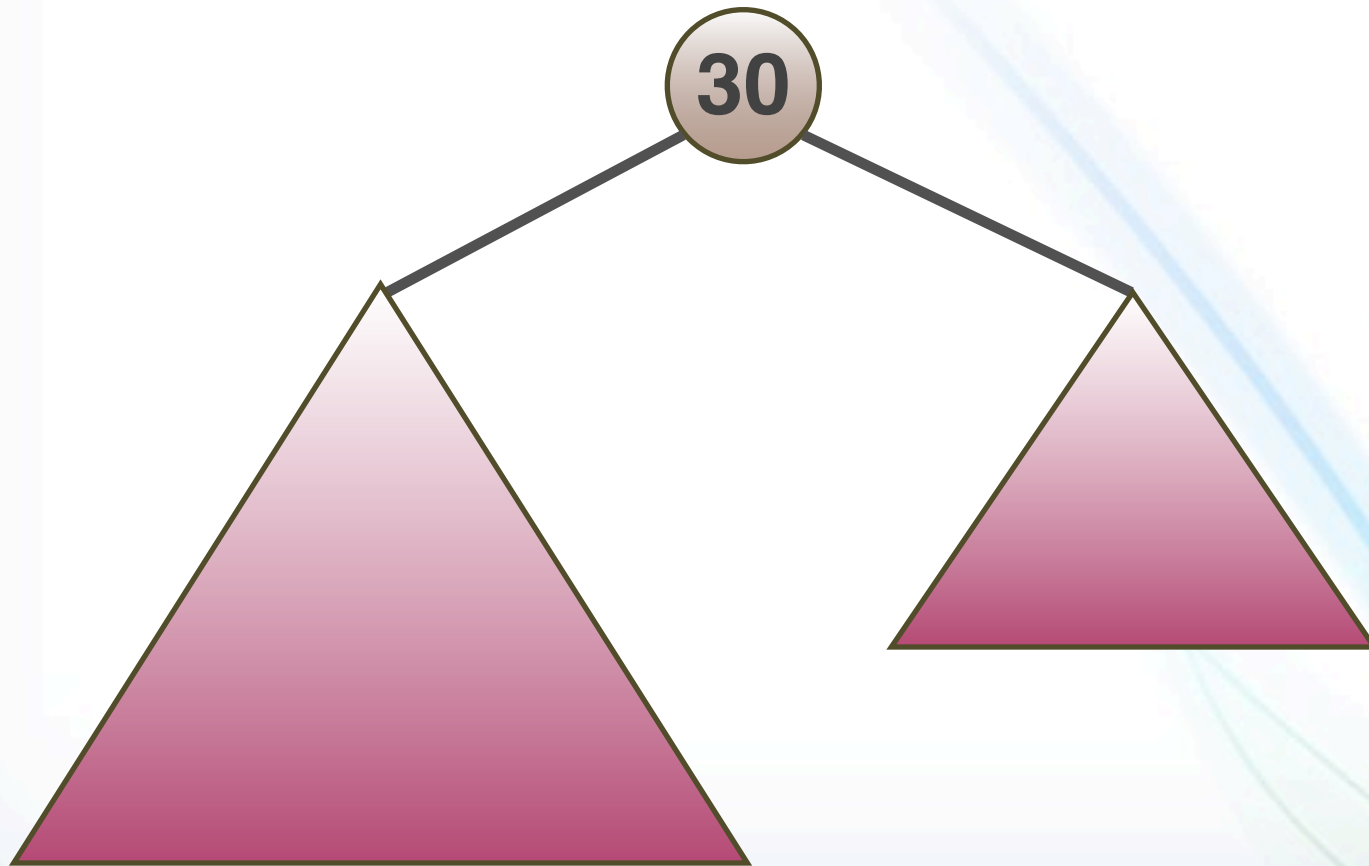
簡単のために、部分木を三角で省略する。



予備知識：部分木の表現

7

簡単のために、部分木を三角で省略する。



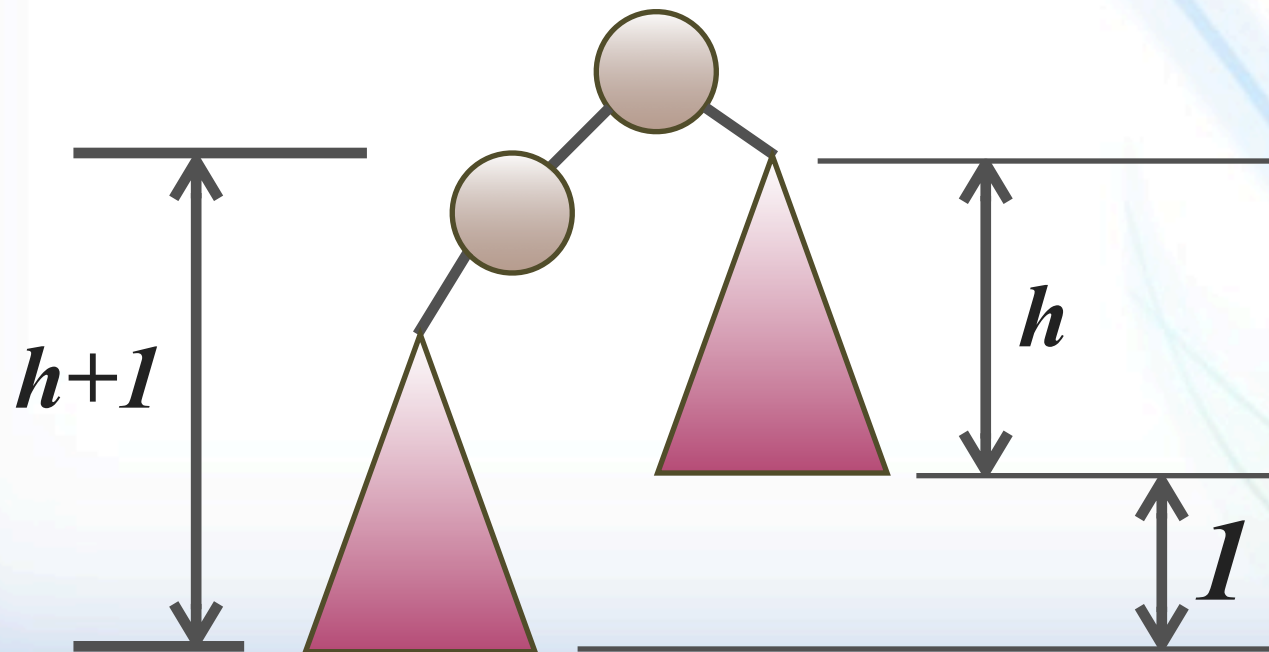
AVL木 (AVL Tree)

8

1962年、Adel'son-Vel'skii と Landisが考案。

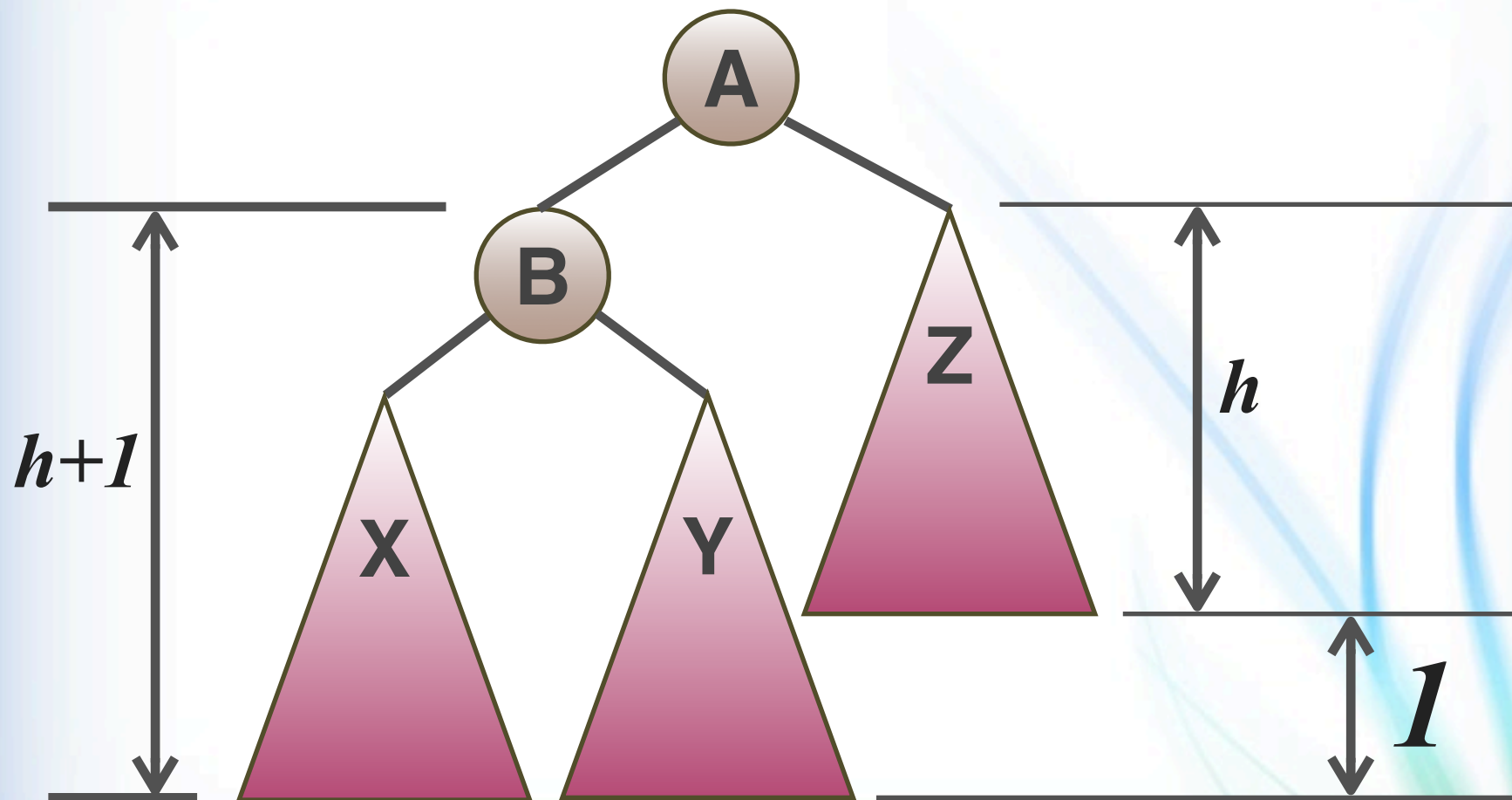
制約条件

- 二分探索木（「左は小さく、右は大きい」）
- 左部分木の高さと右部分木の高さの差が1以内であること。



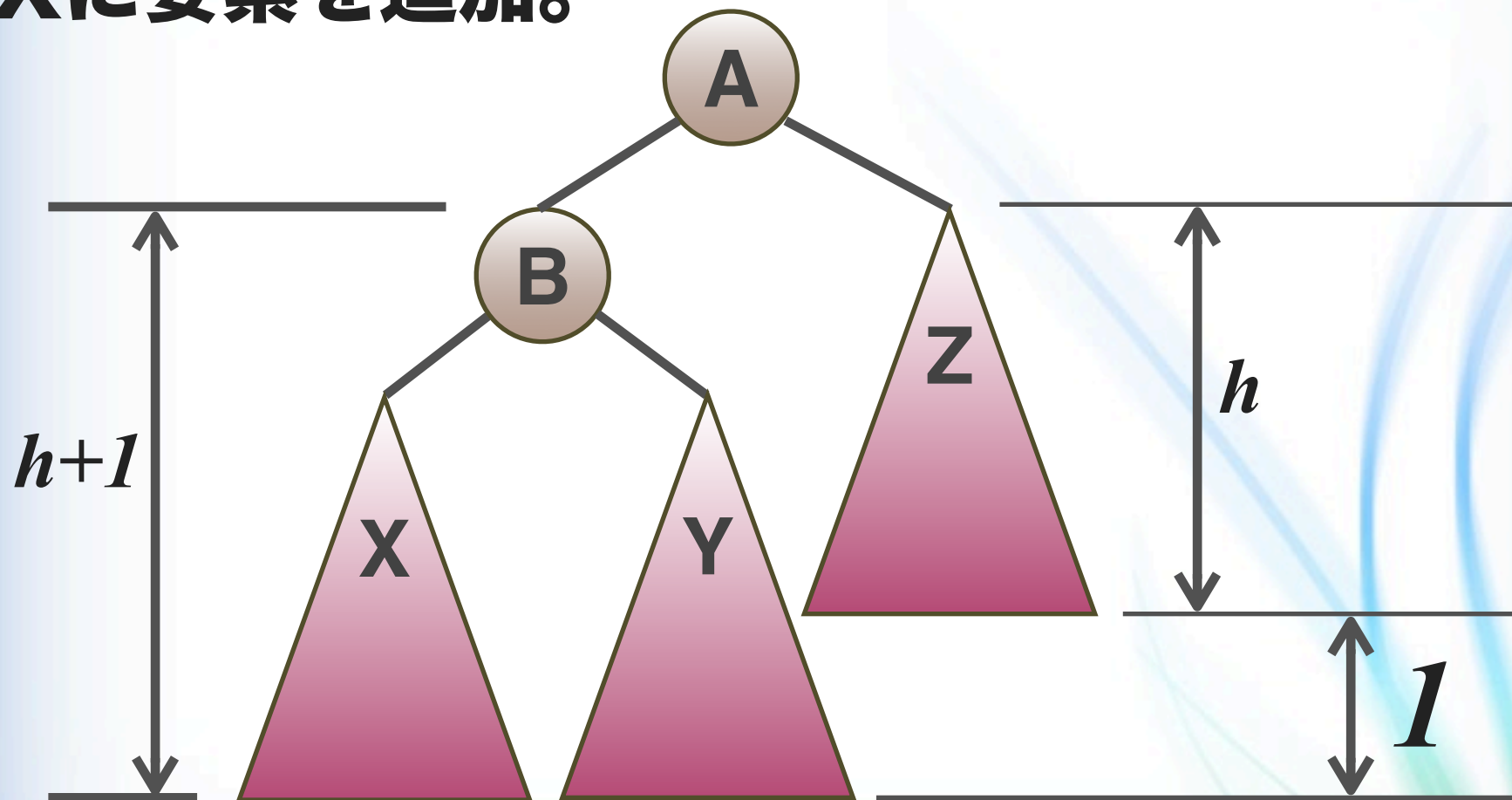
AVL木の基本形（平衡状態）

9



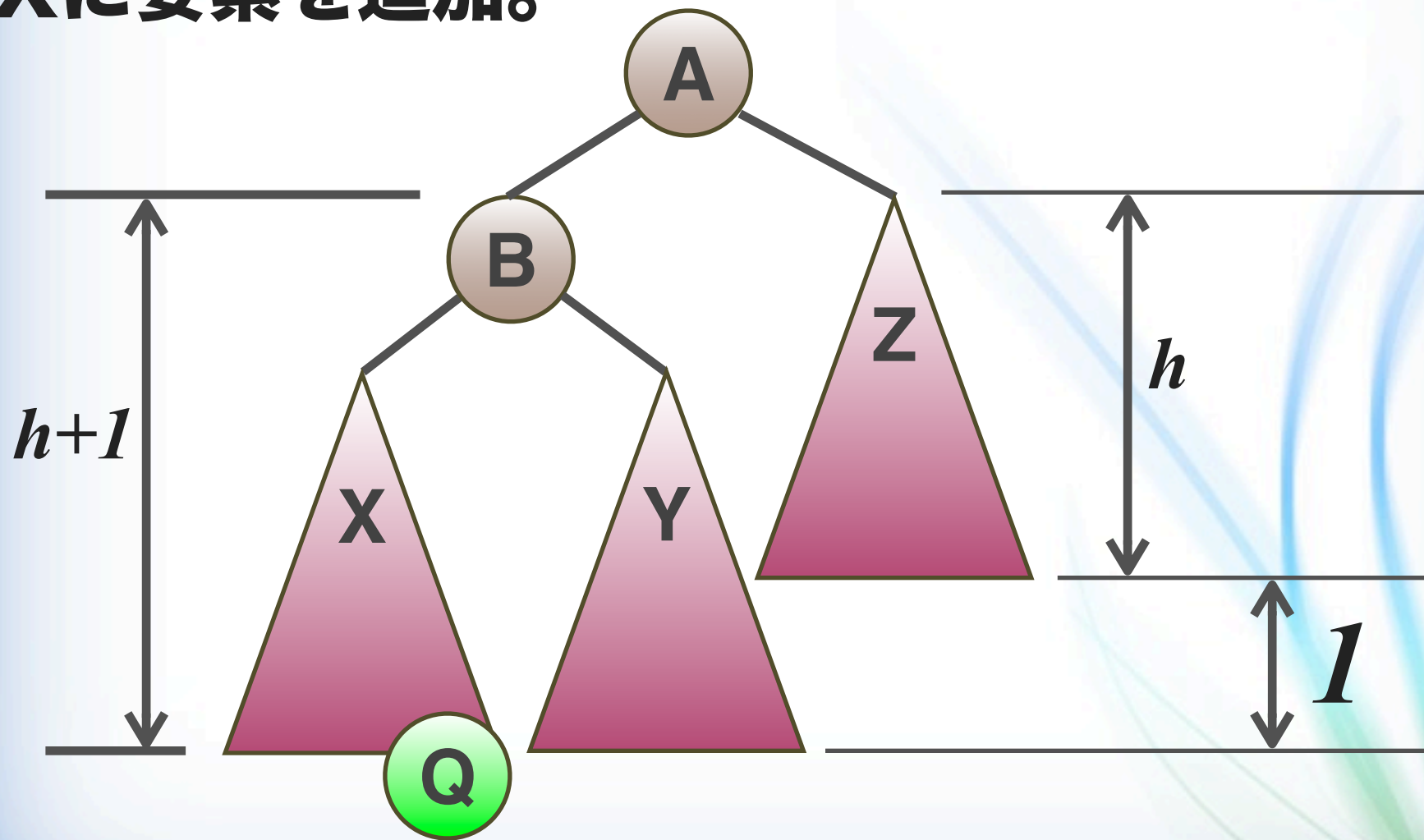
AVL木の追加処理（左側への追加） 10

Xに要素を追加。



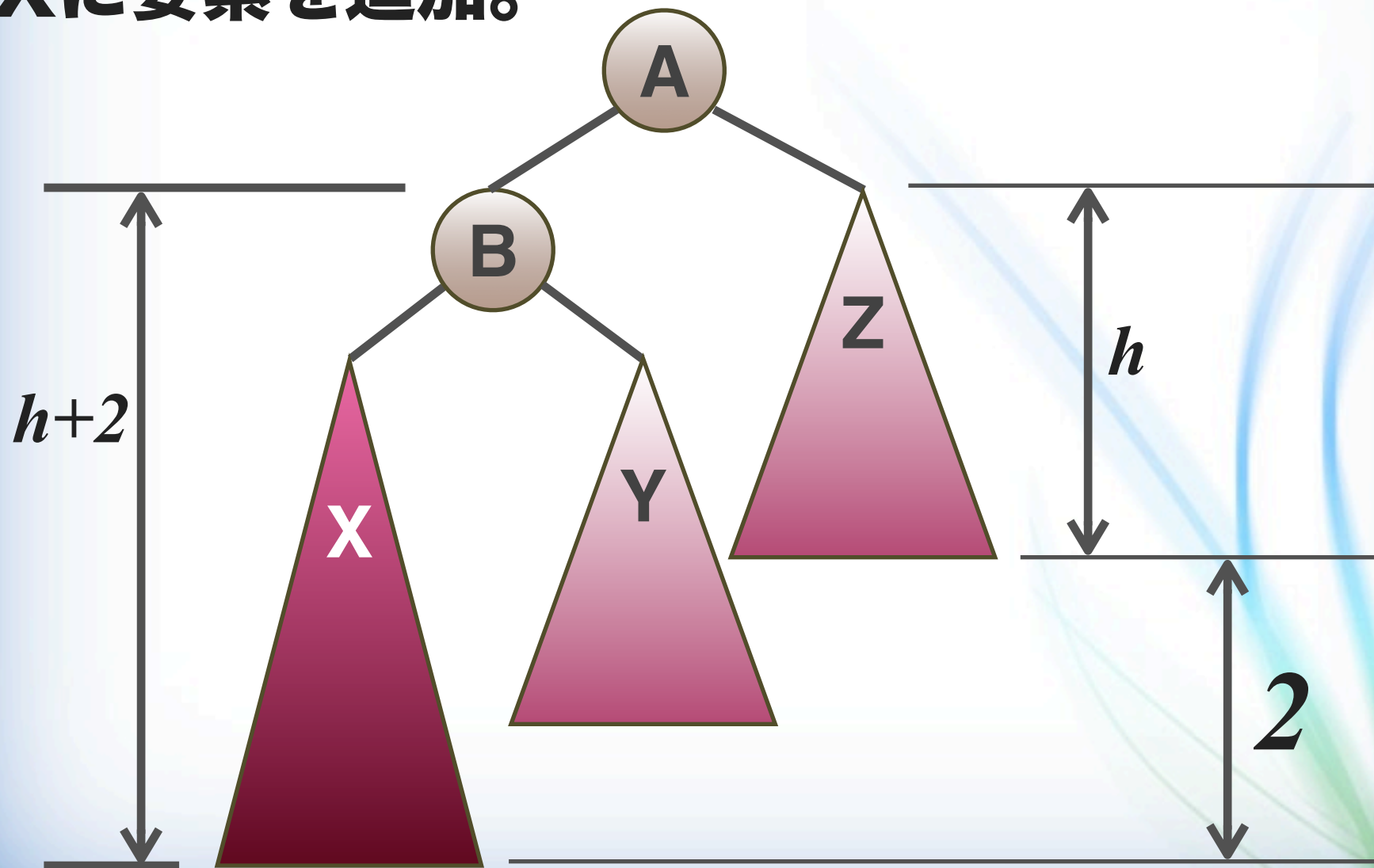
AVL木の追加処理（左側への追加） 10

Xに要素を追加。



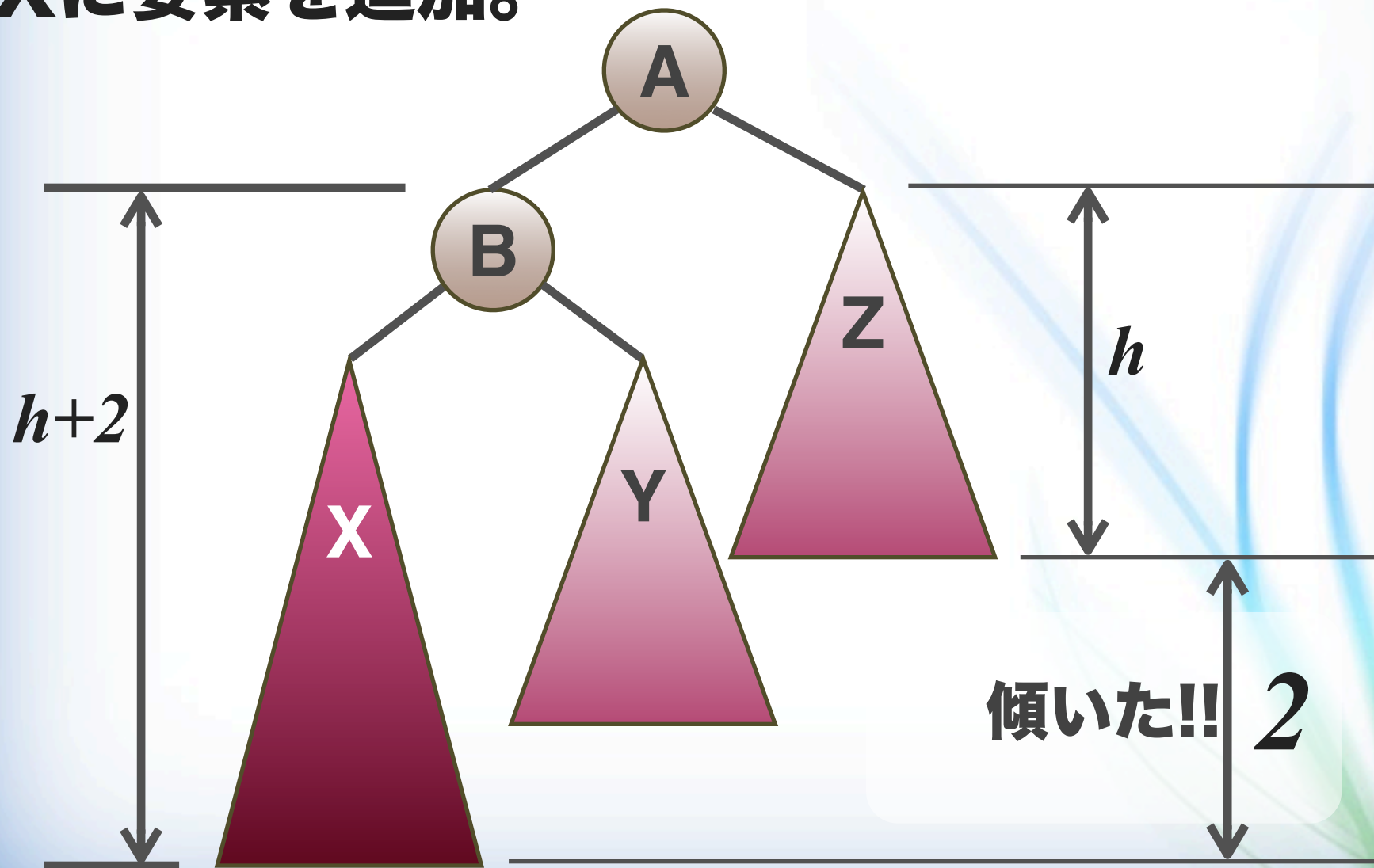
AVL木の追加処理（左側への追加） 11

Xに要素を追加。



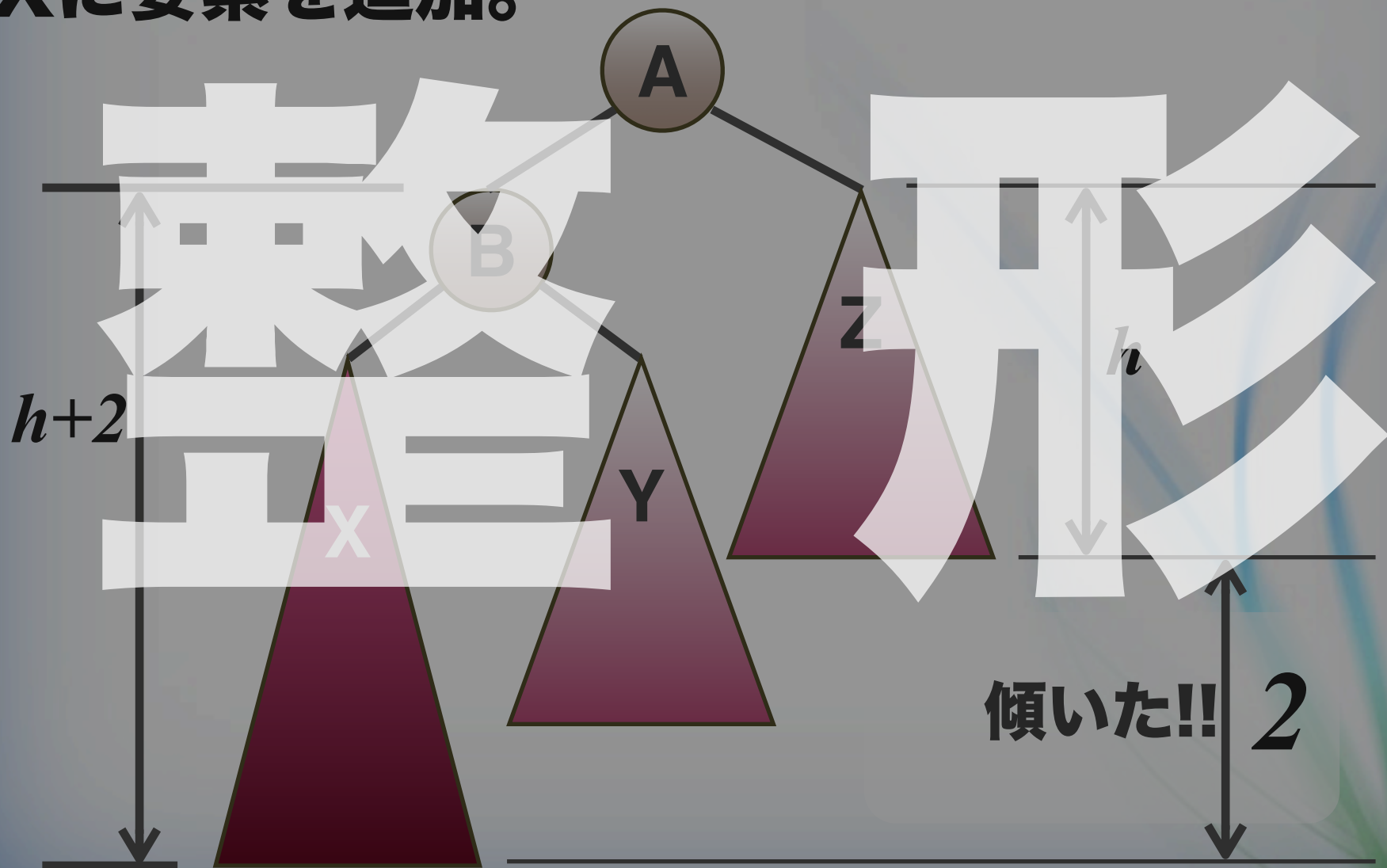
AVL木の追加処理（左側への追加） 11

Xに要素を追加。



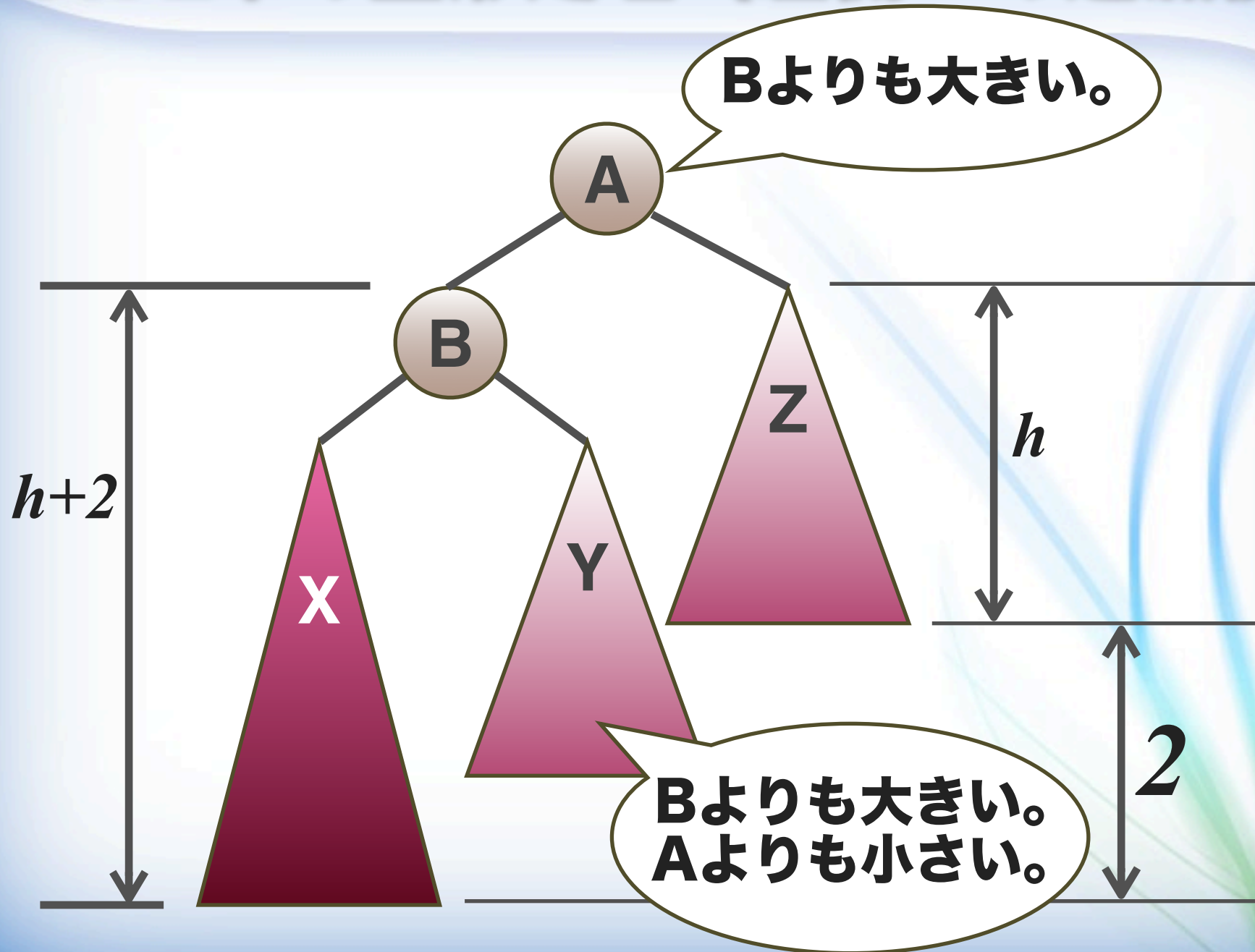
AVL木の追加処理（左側への追加） 11

Xに要素を追加。

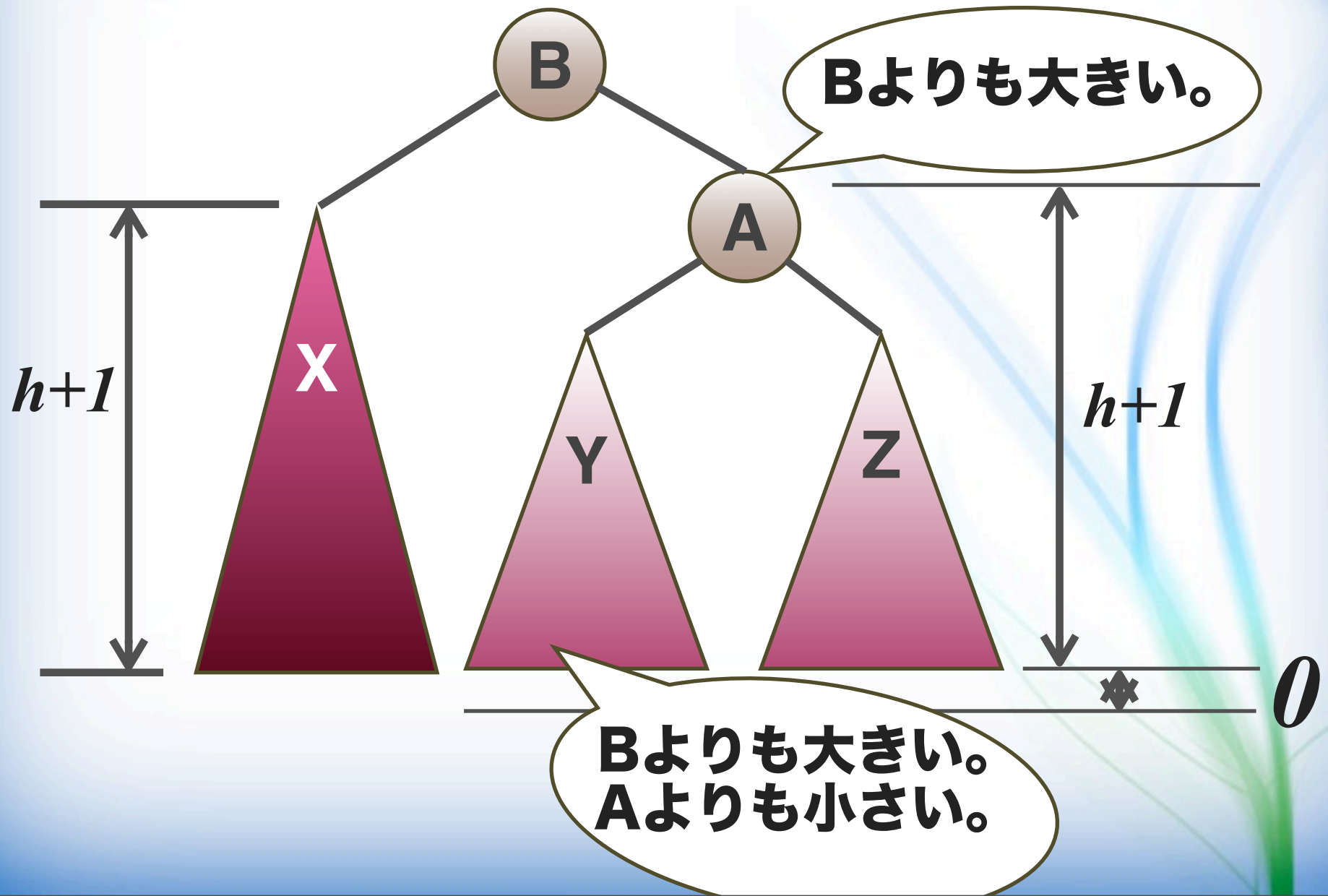


AVL木の整形処理（左側への追加後）

12



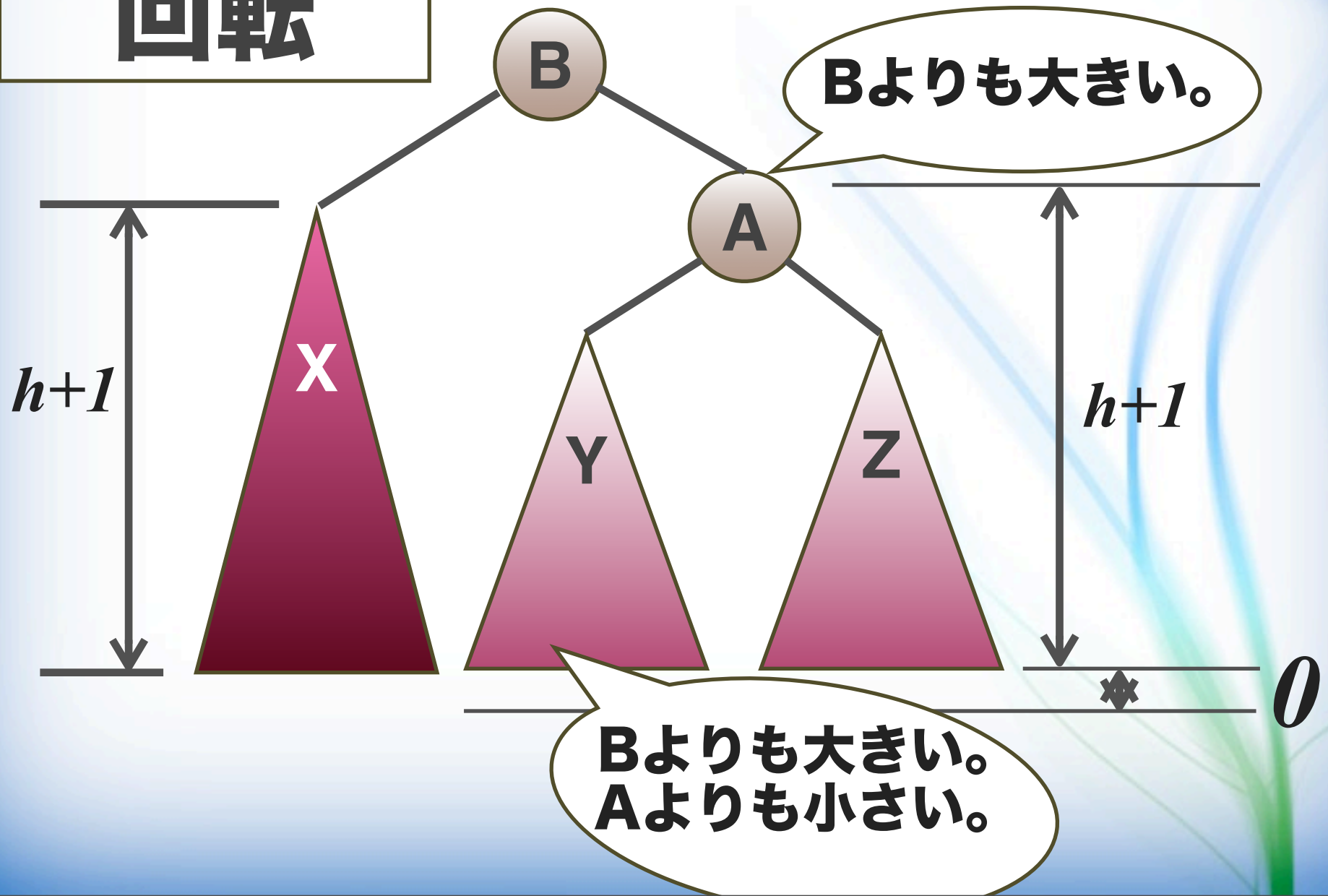
13



AVL木の整形処理（左側への追加後）

13

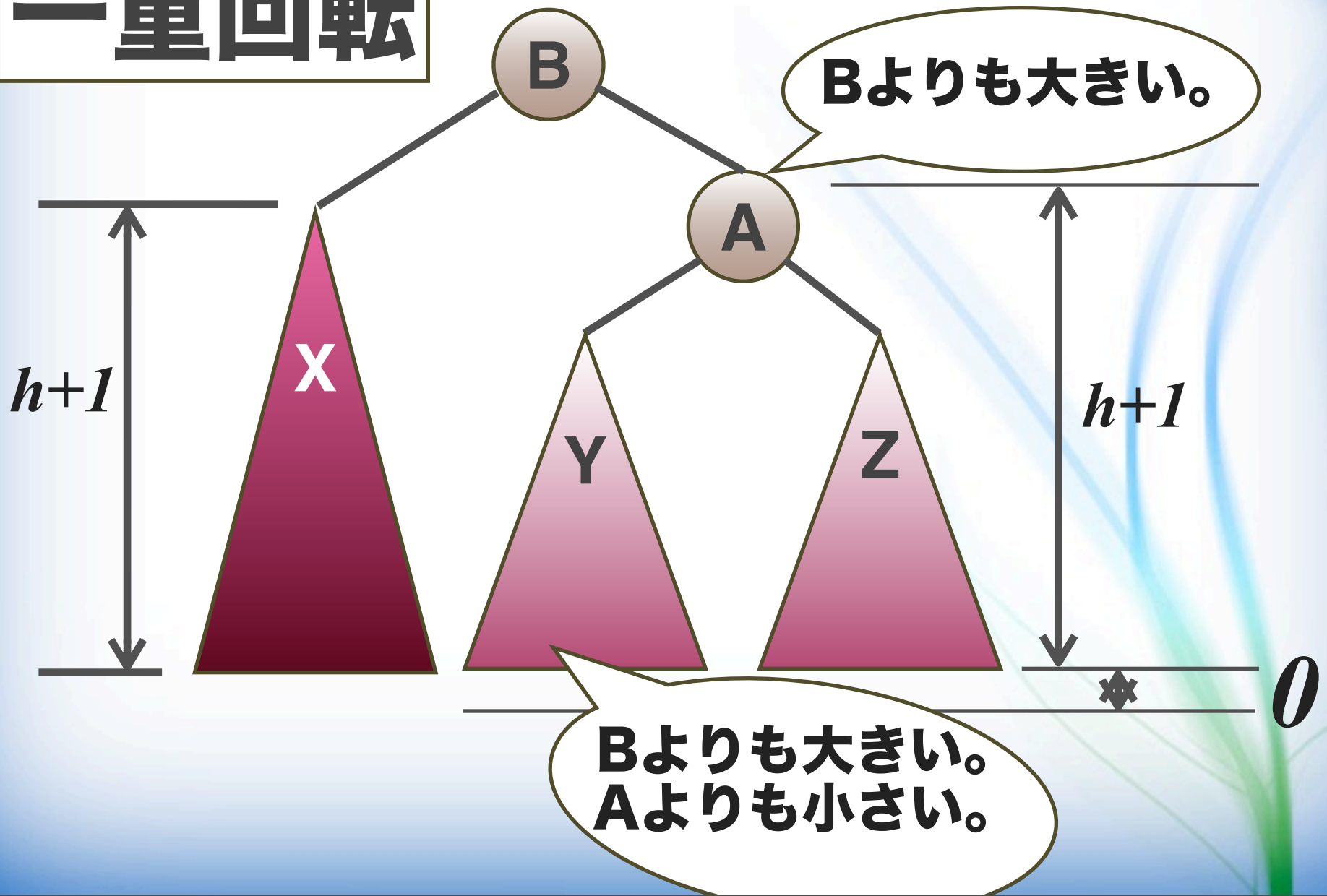
回転



AVL木の整形処理（左側への追加後）

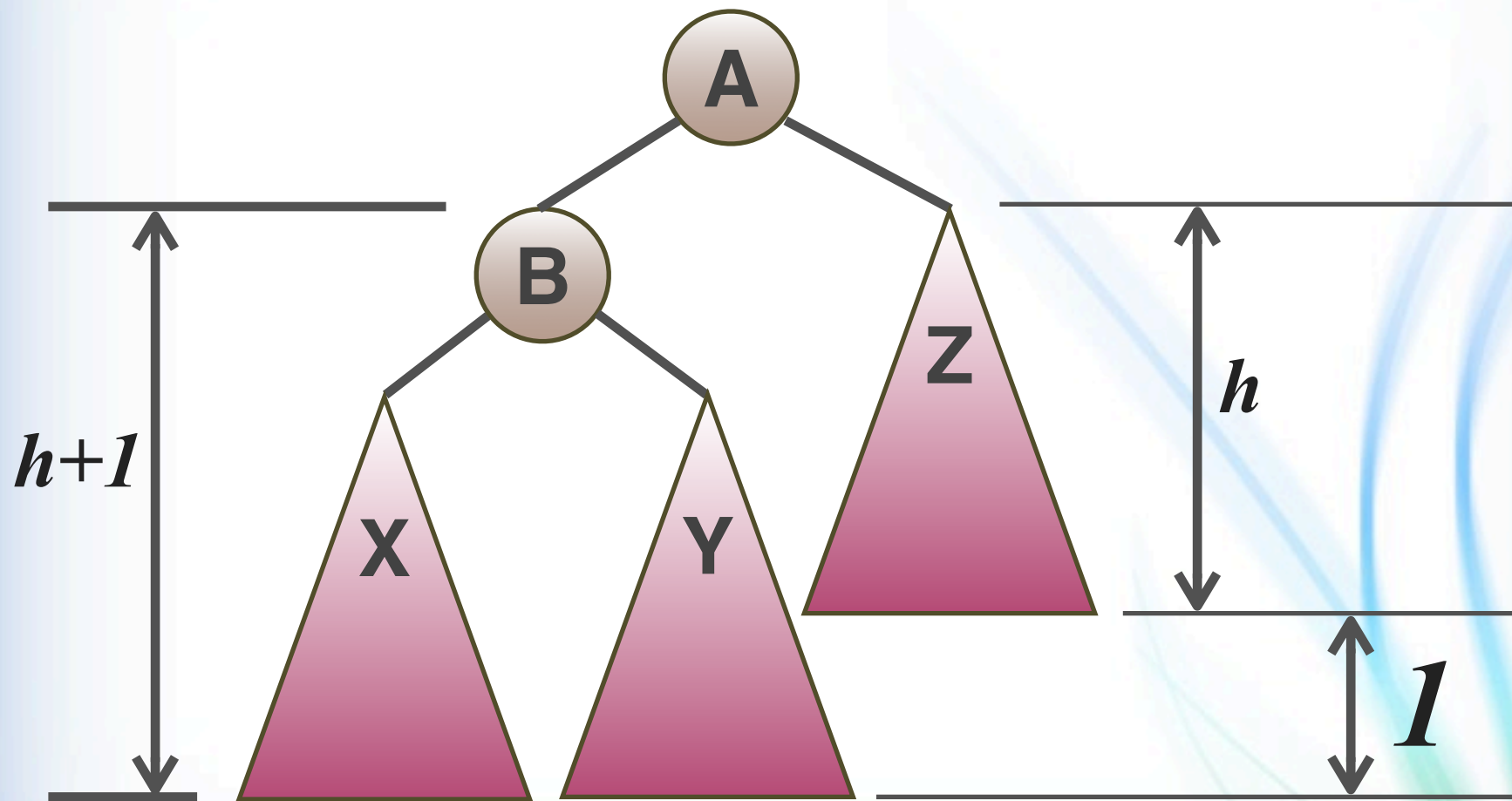
13

一重回転



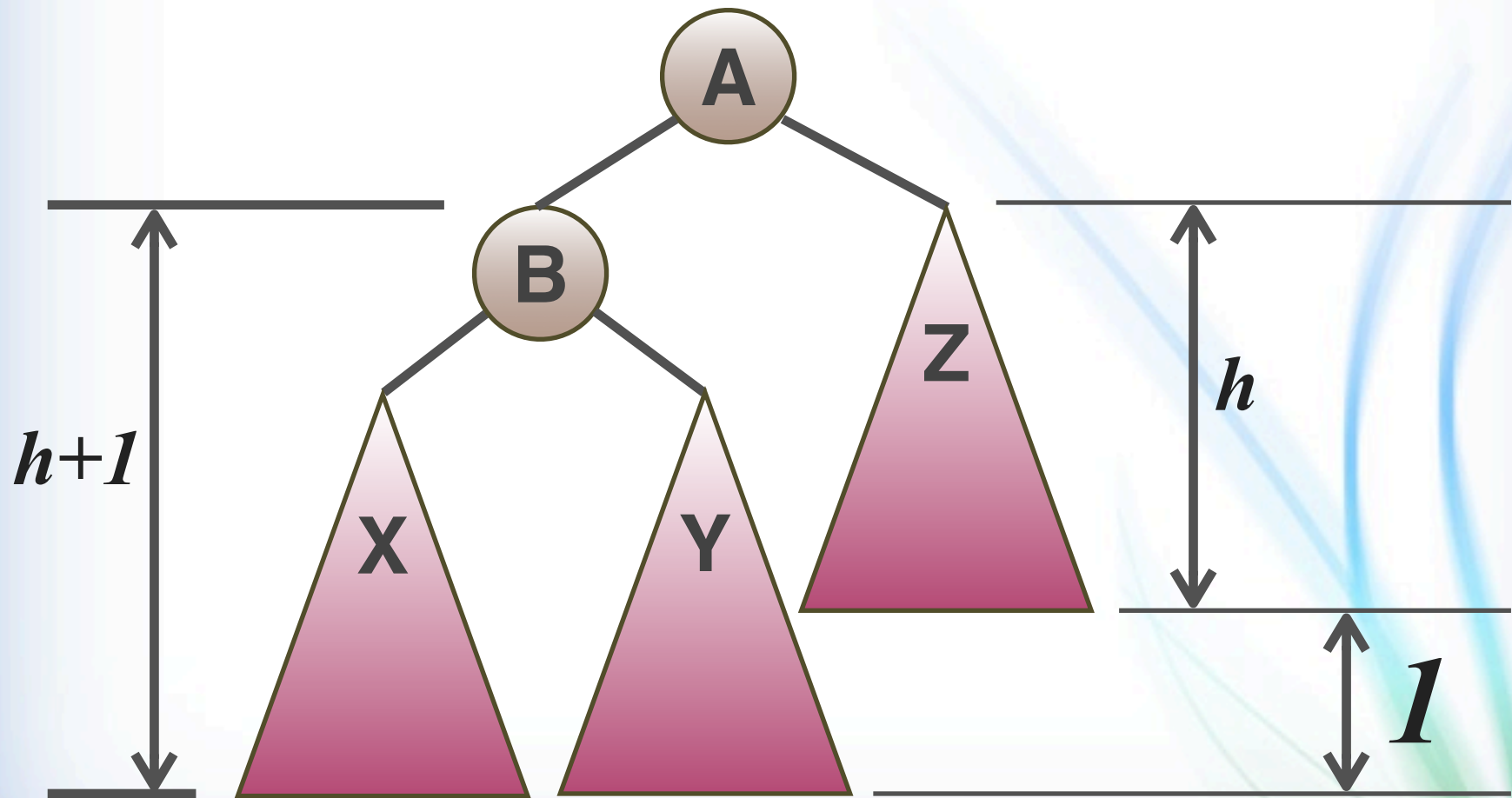
AVL木の基本形（平衡状態）

14



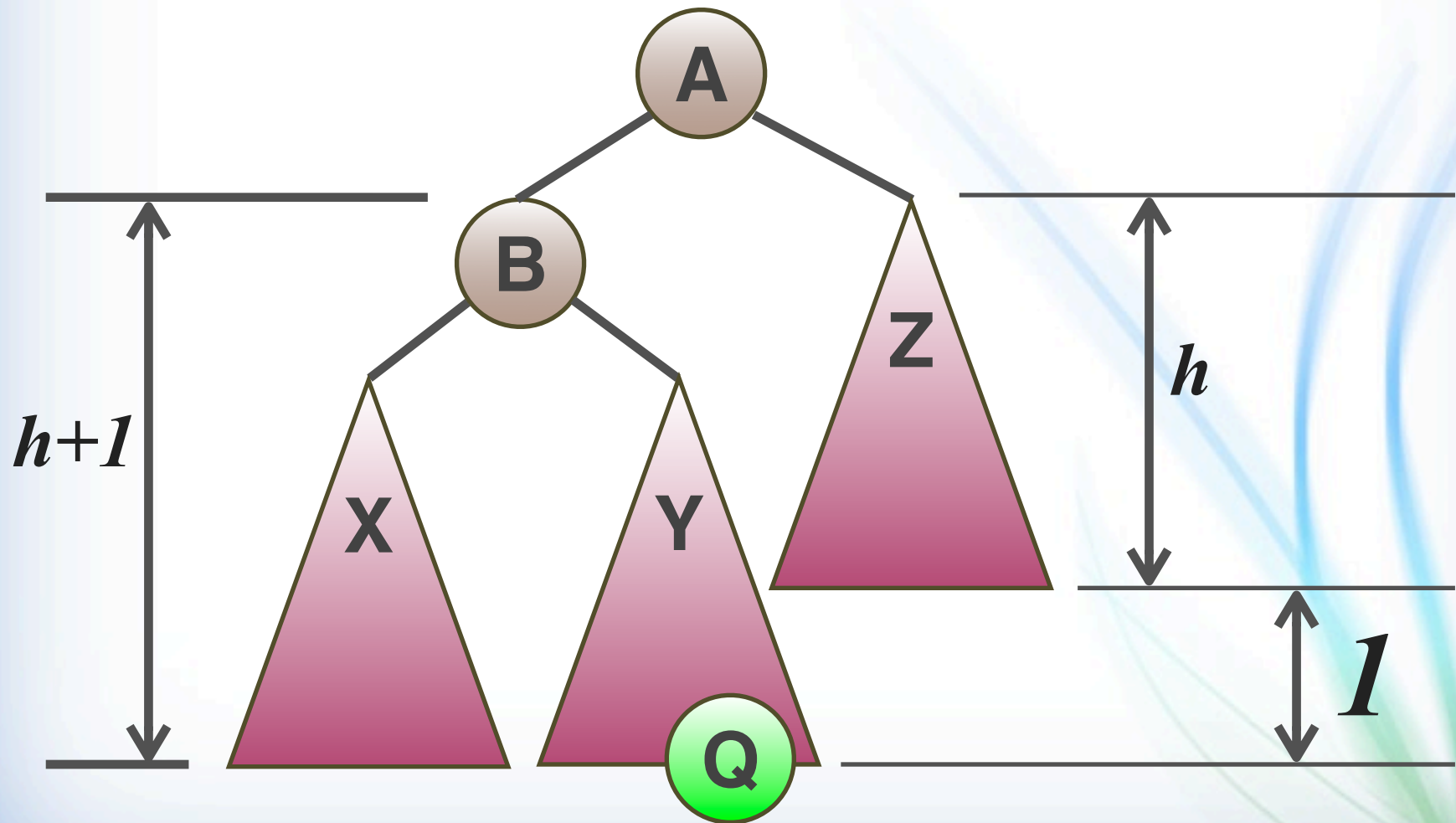
AVL木の追加処理（右側への追加） 15

Yに要素を追加。



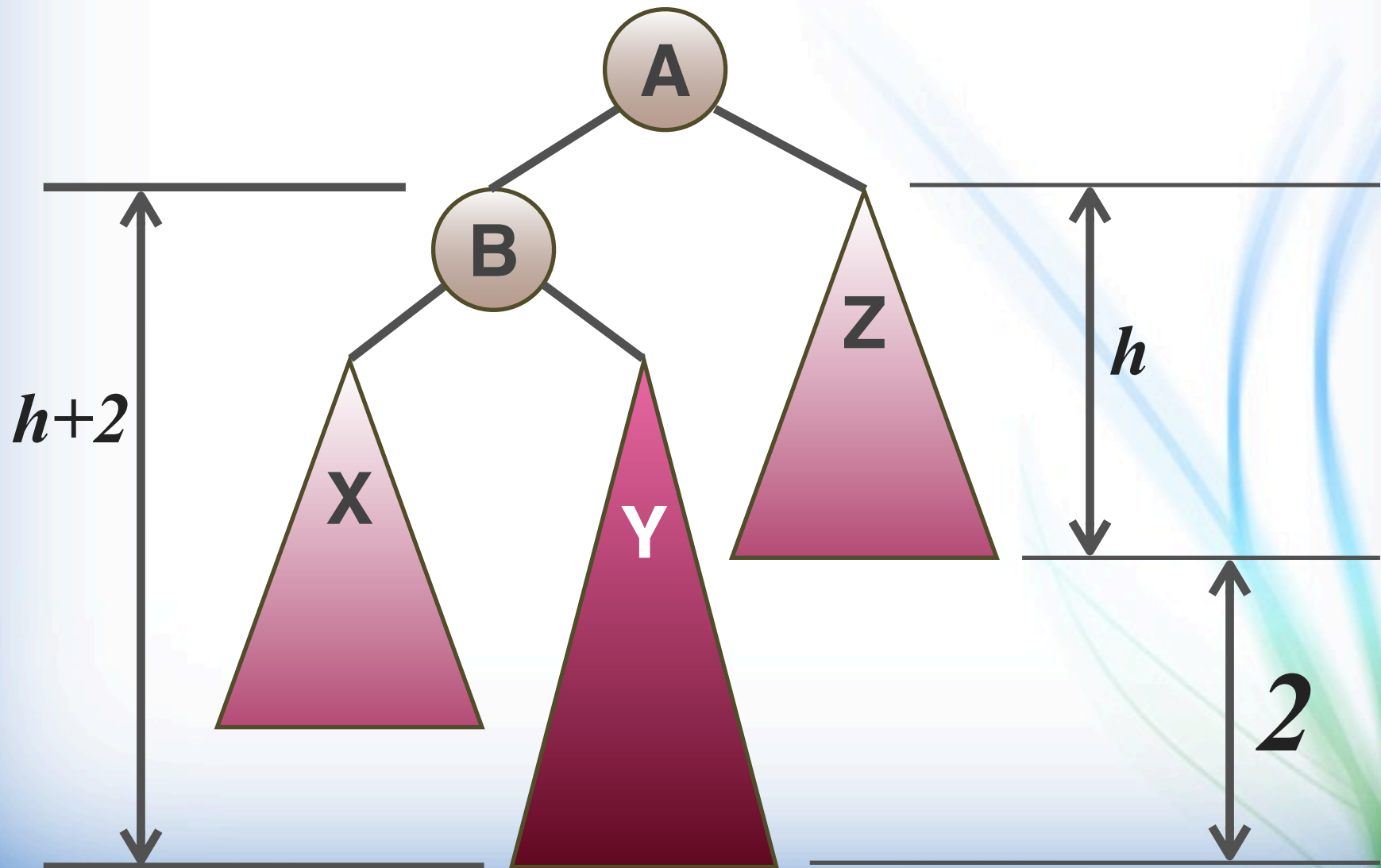
AVL木の追加処理（右側への追加） 15

Yに要素を追加。



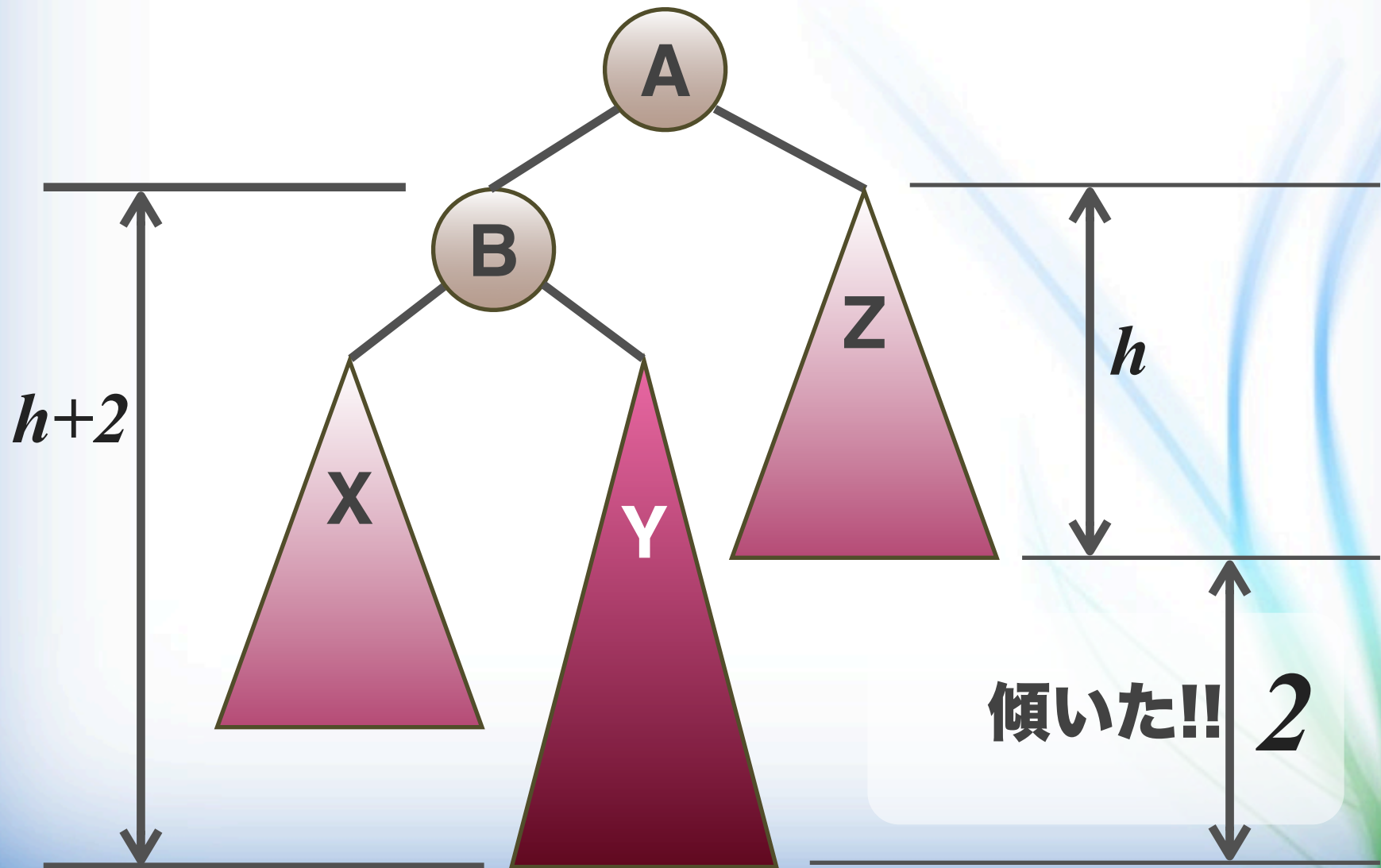
AVL木の追加処理（右側への追加） 16

Yに要素を追加。



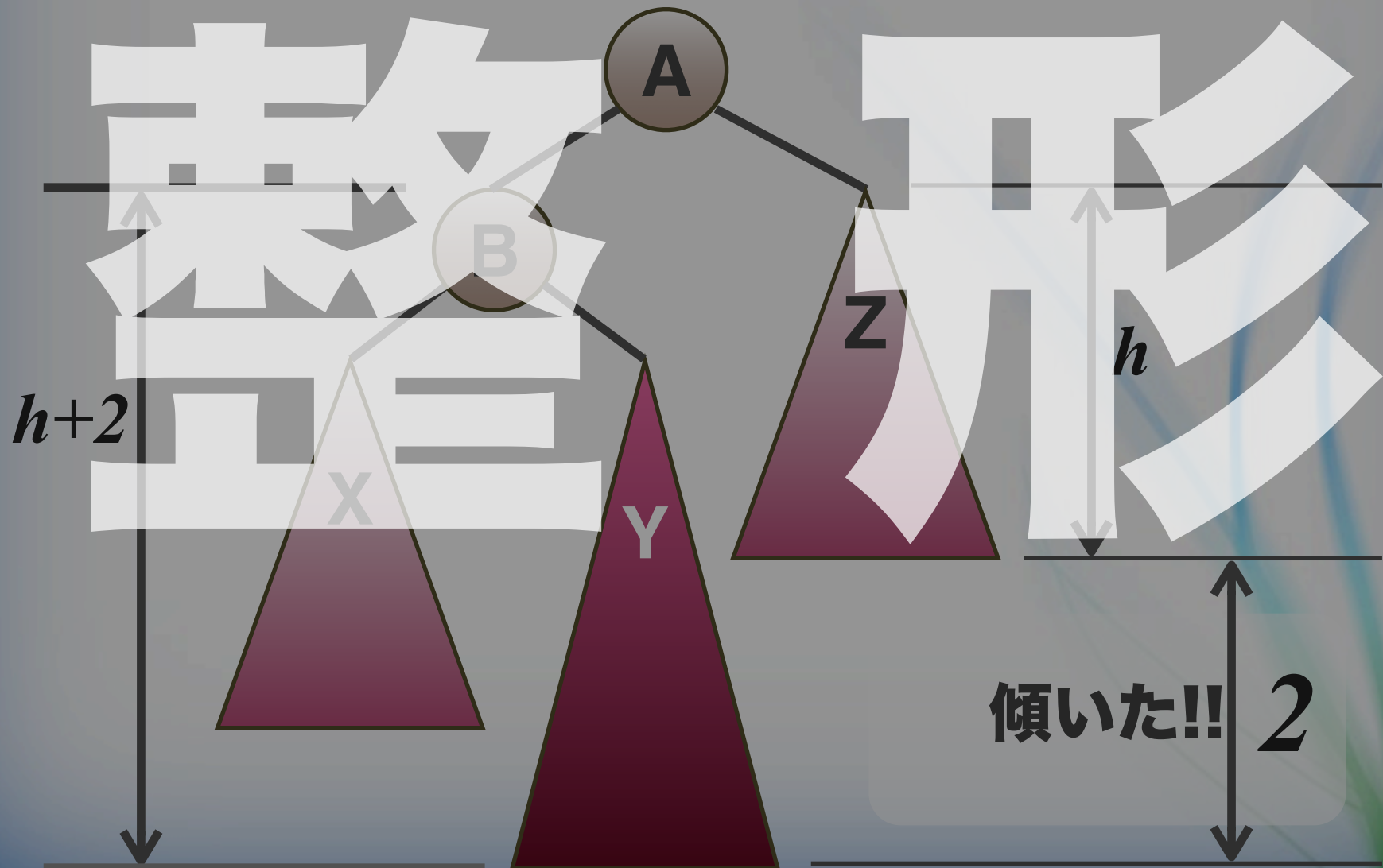
AVL木の追加処理（右側への追加） 16

Yに要素を追加。



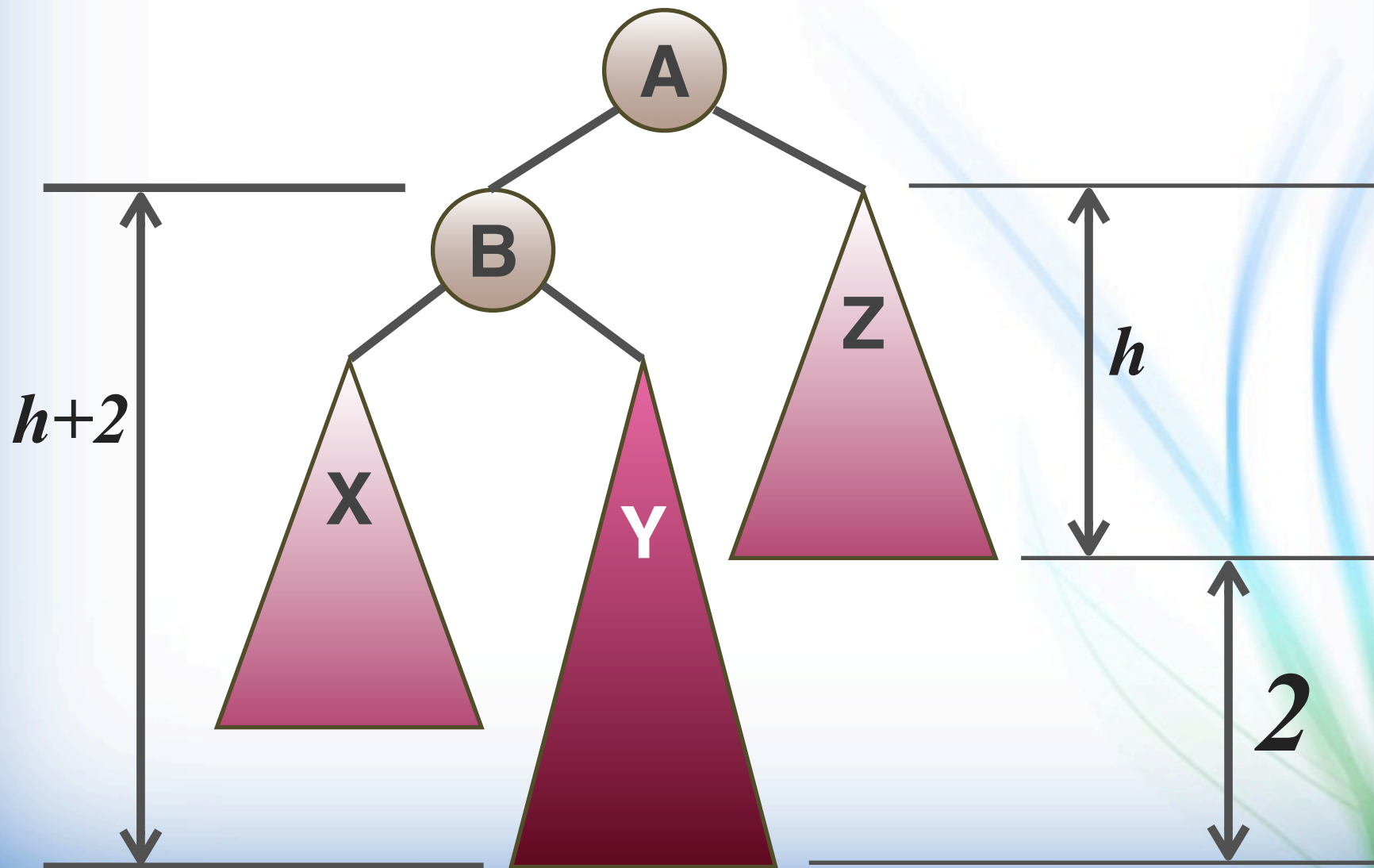
AVL木の追加処理（右側への追加） 16

Yに要素を追加。



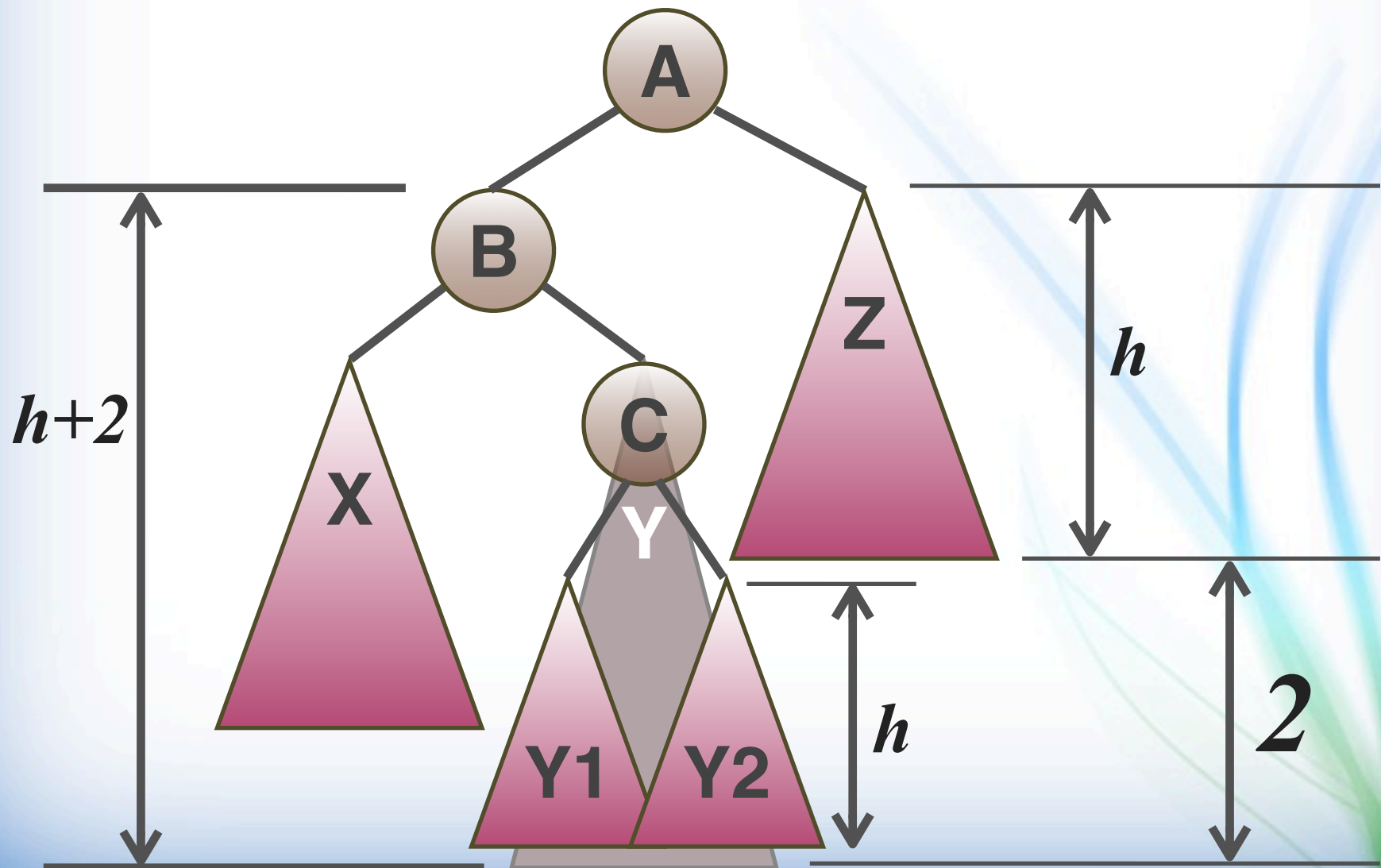
AVL木の追加処理（右側への追加） 17

Yを分解してみる。

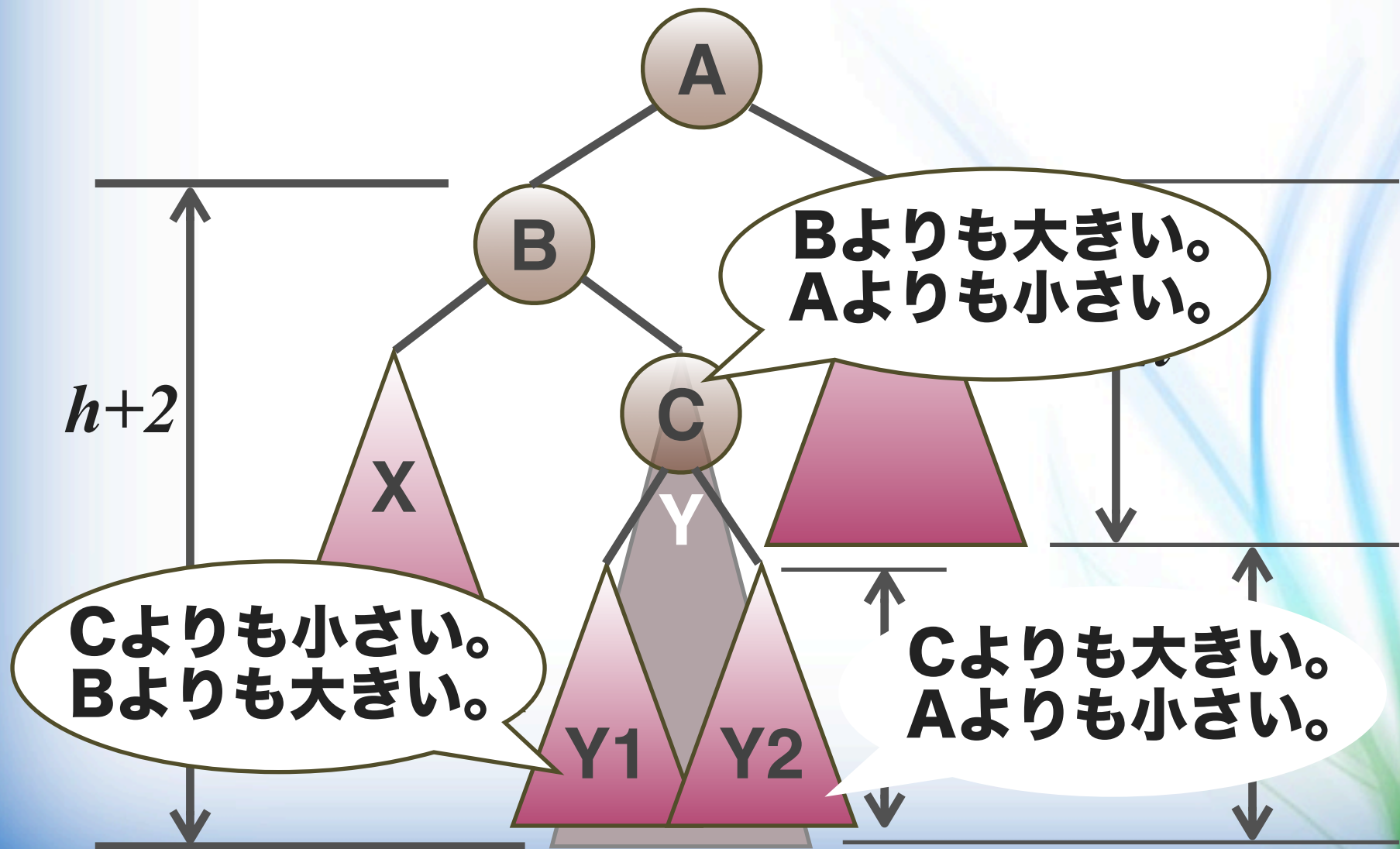


AVL木の追加処理（右側への追加） 18

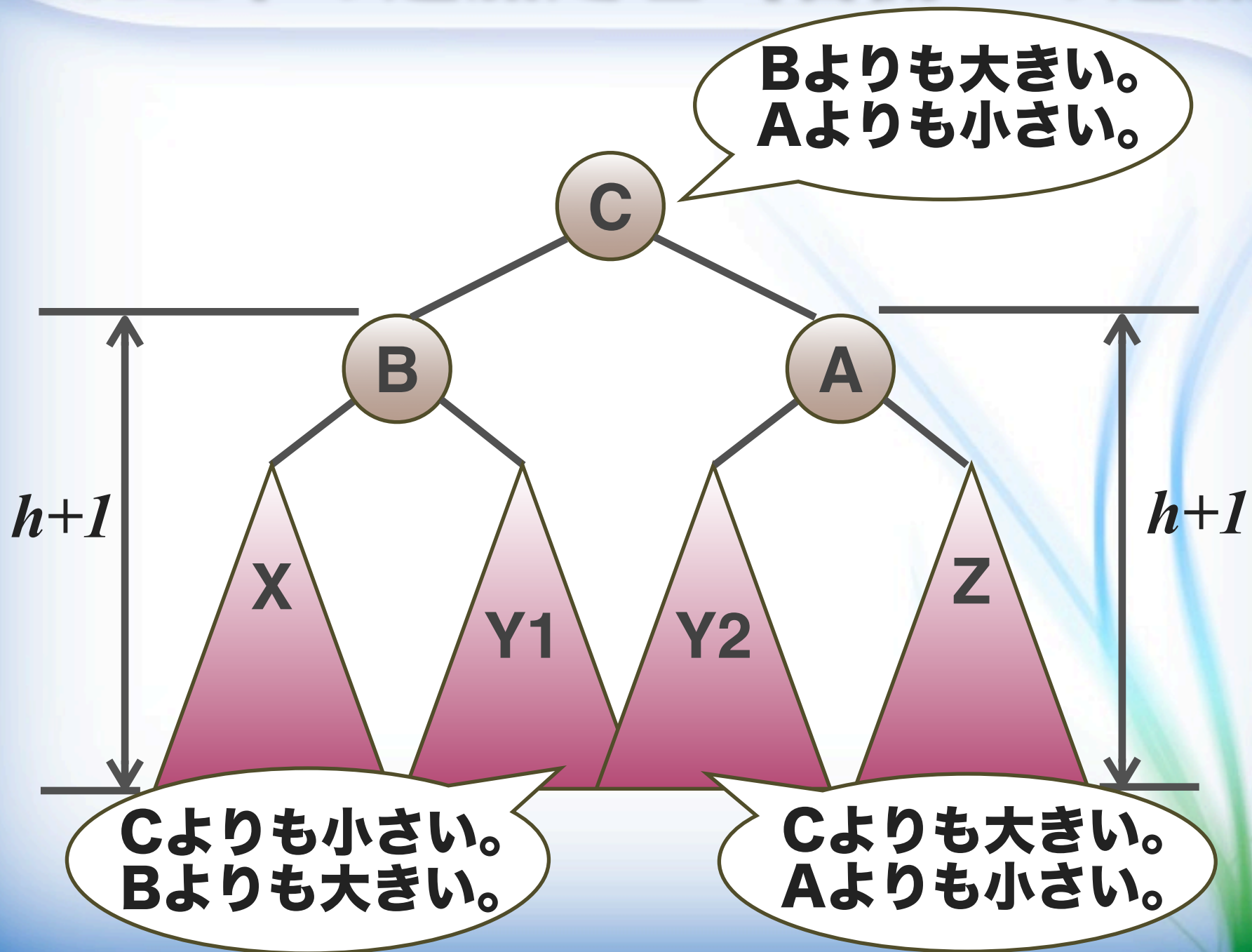
Yを分解してみる。



AVL木の追加処理（右側への追加） 19



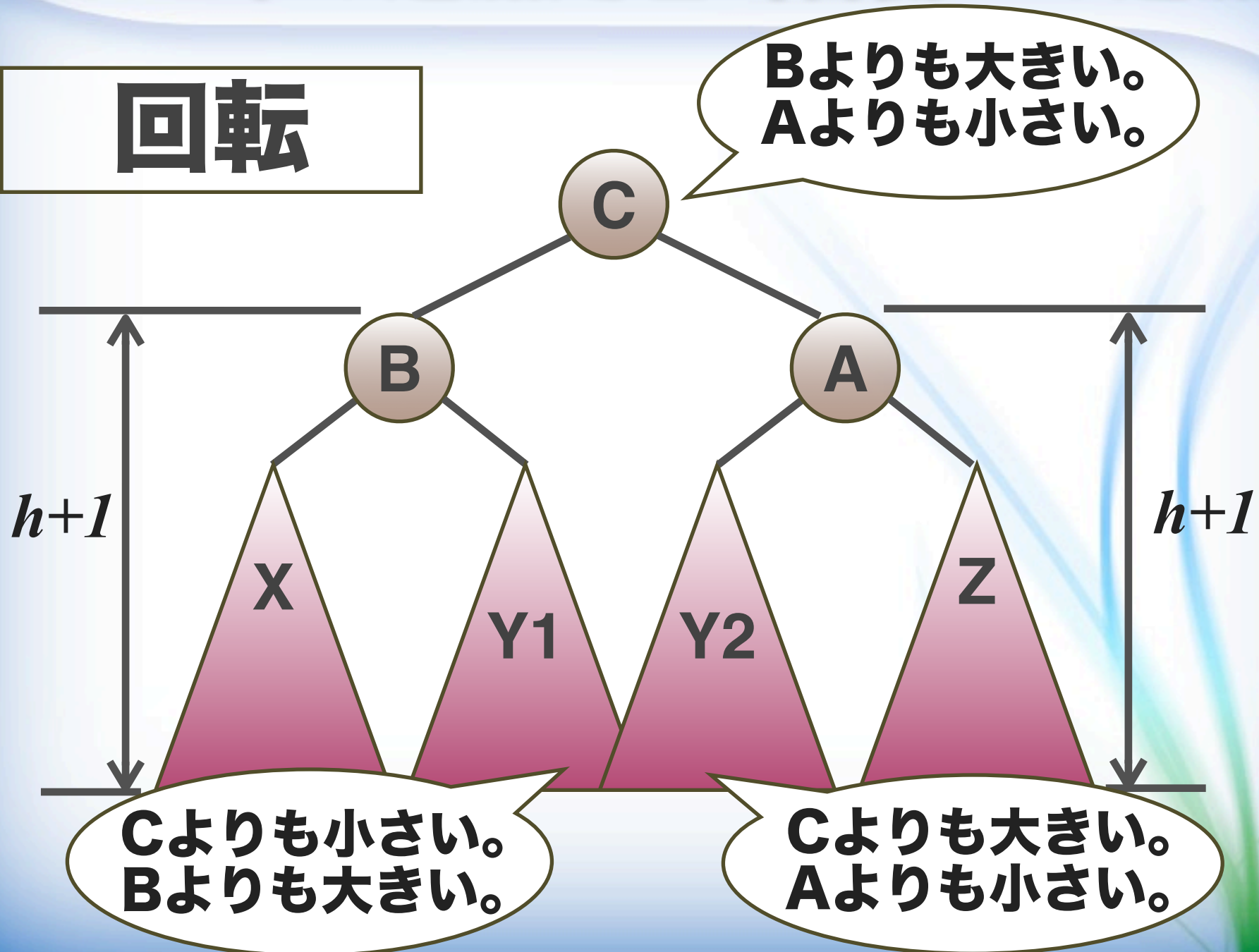
AVL木の追加処理（右側への追加） 20



AVL木の追加処理（右側への追加）

20

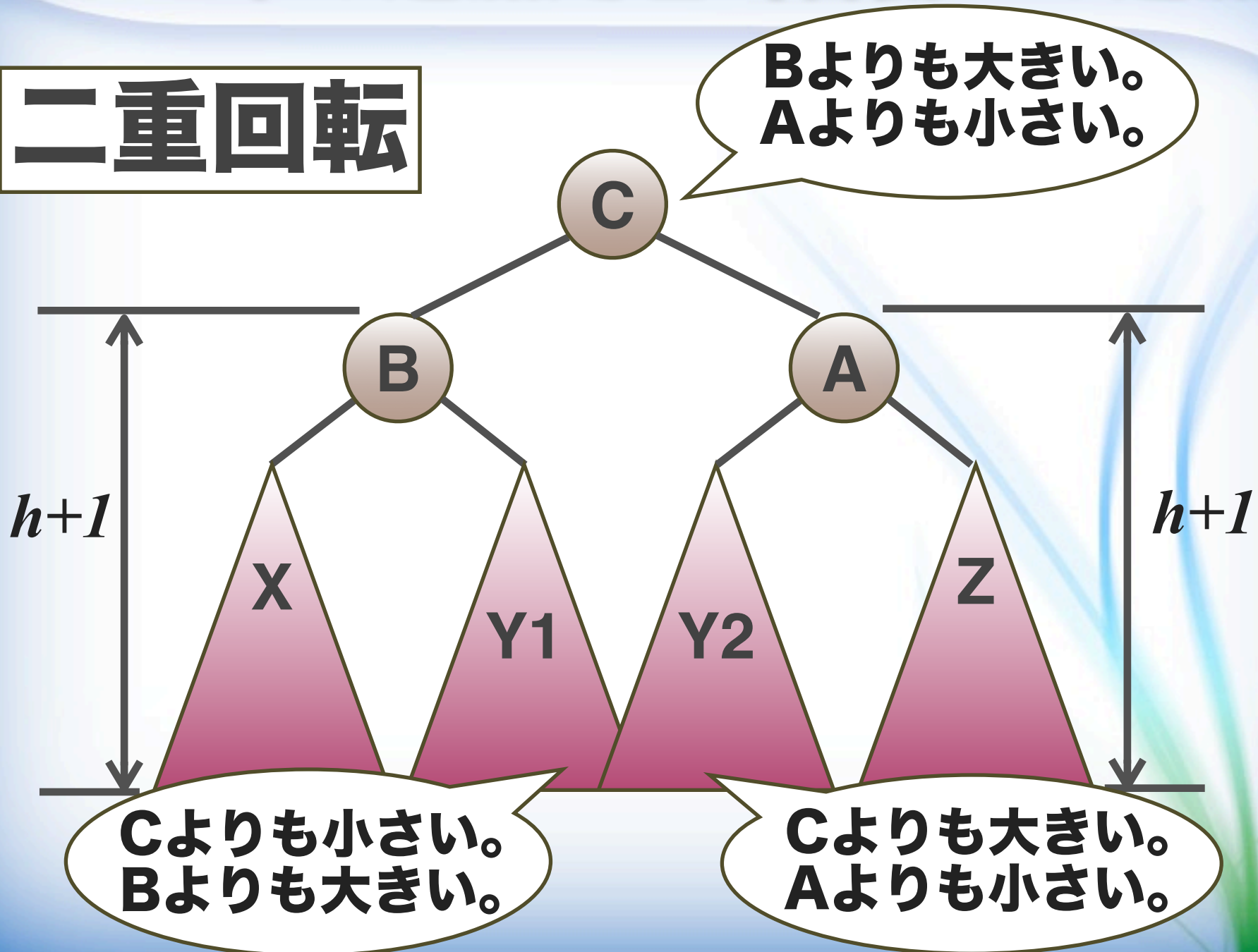
回転



AVL木の追加処理（右側への追加）

20

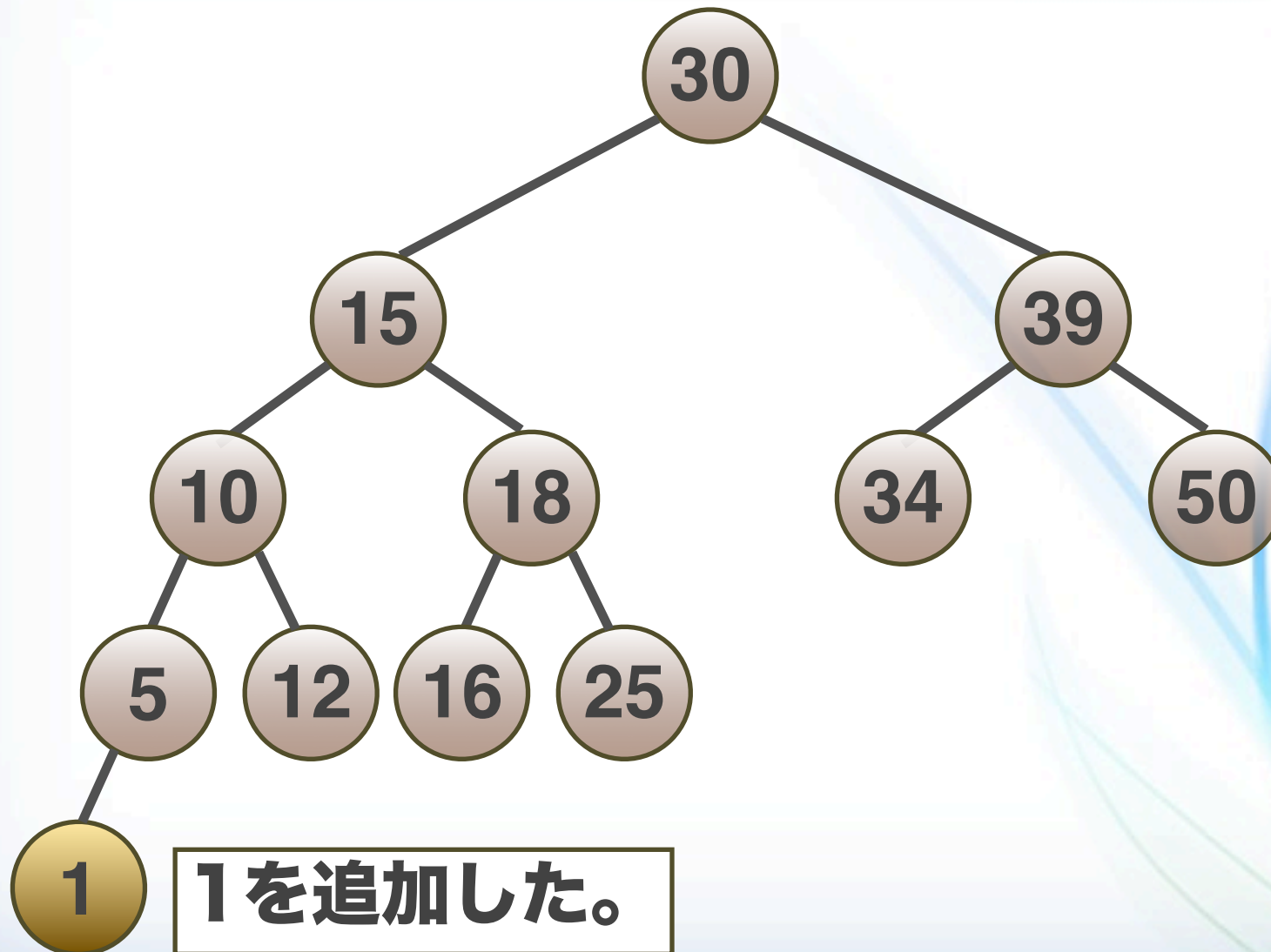
二重回転



AVL木の事例

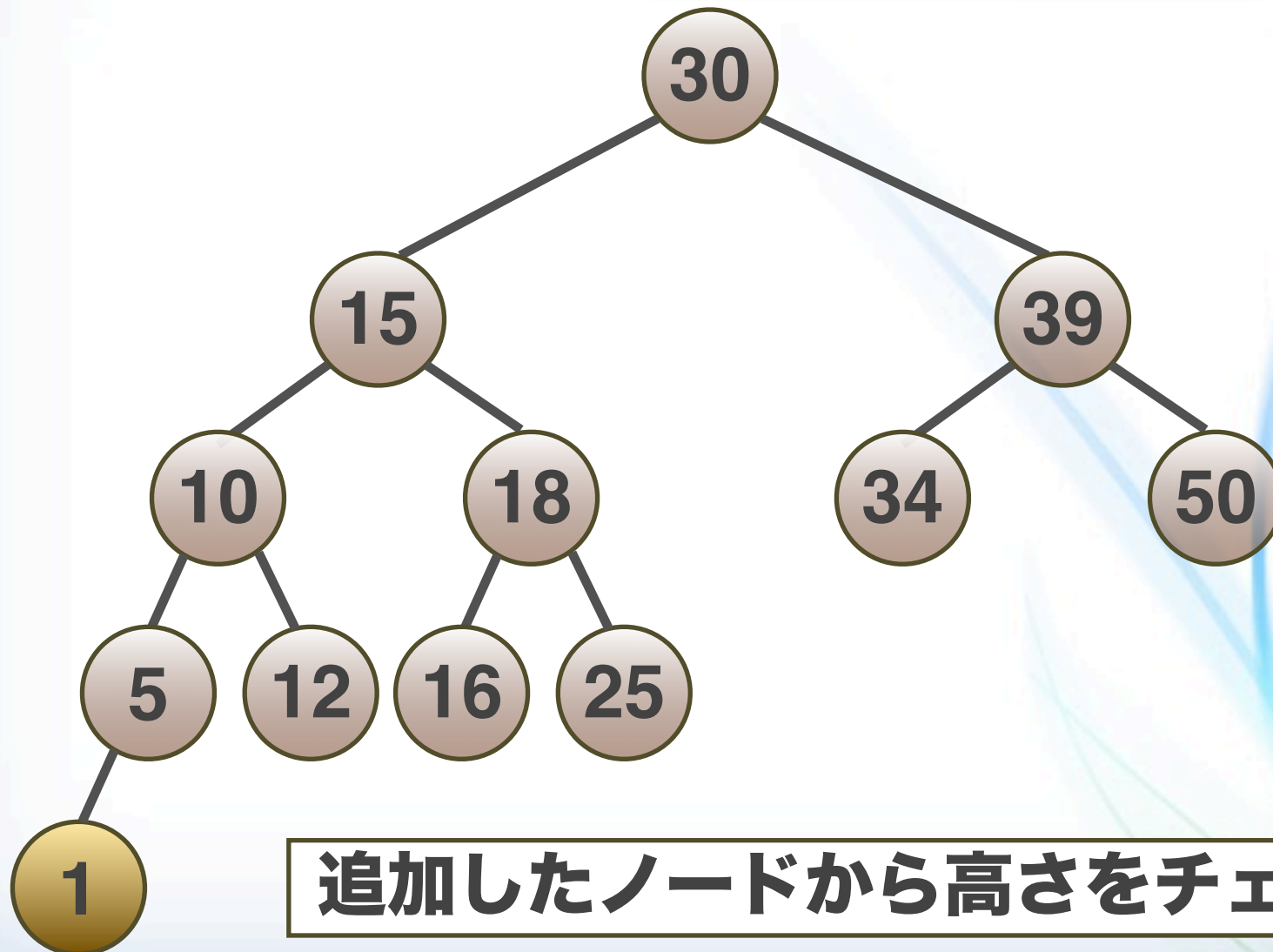
AVL木の事例 1

22



AVL木の実例 1

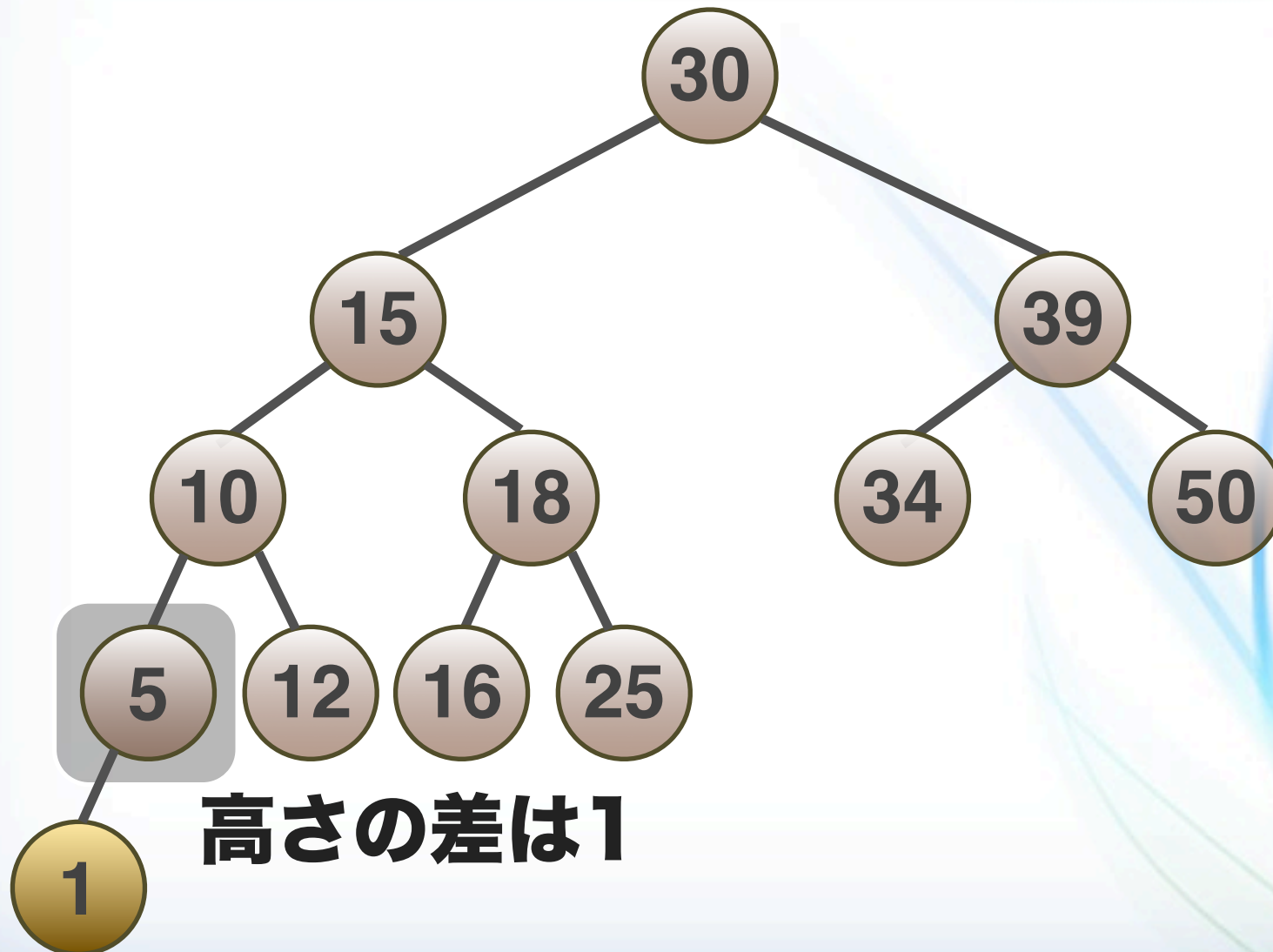
23



追加したノードから高さをチェック。

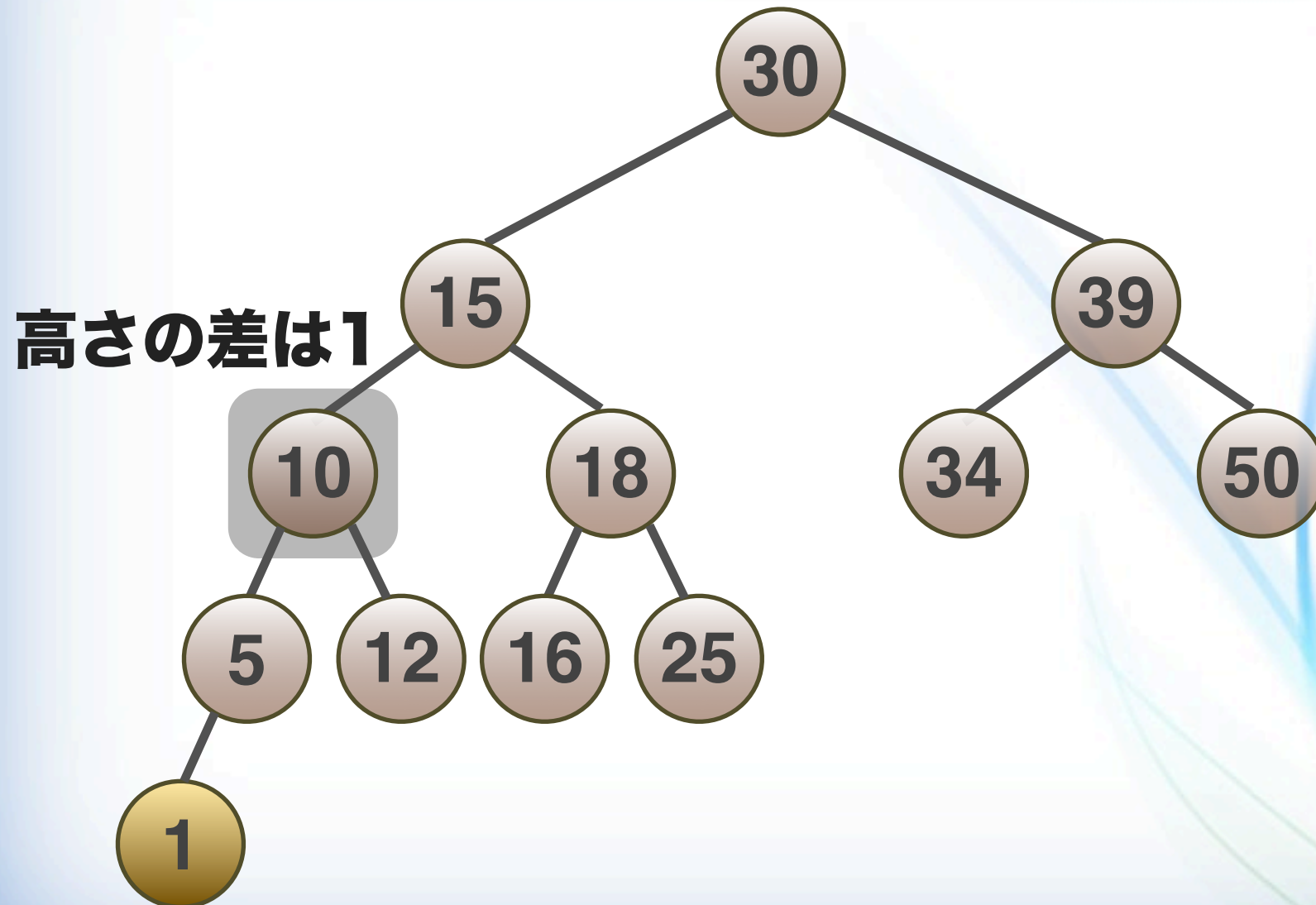
AVL木の実例 1

24



AVL木の事例 1

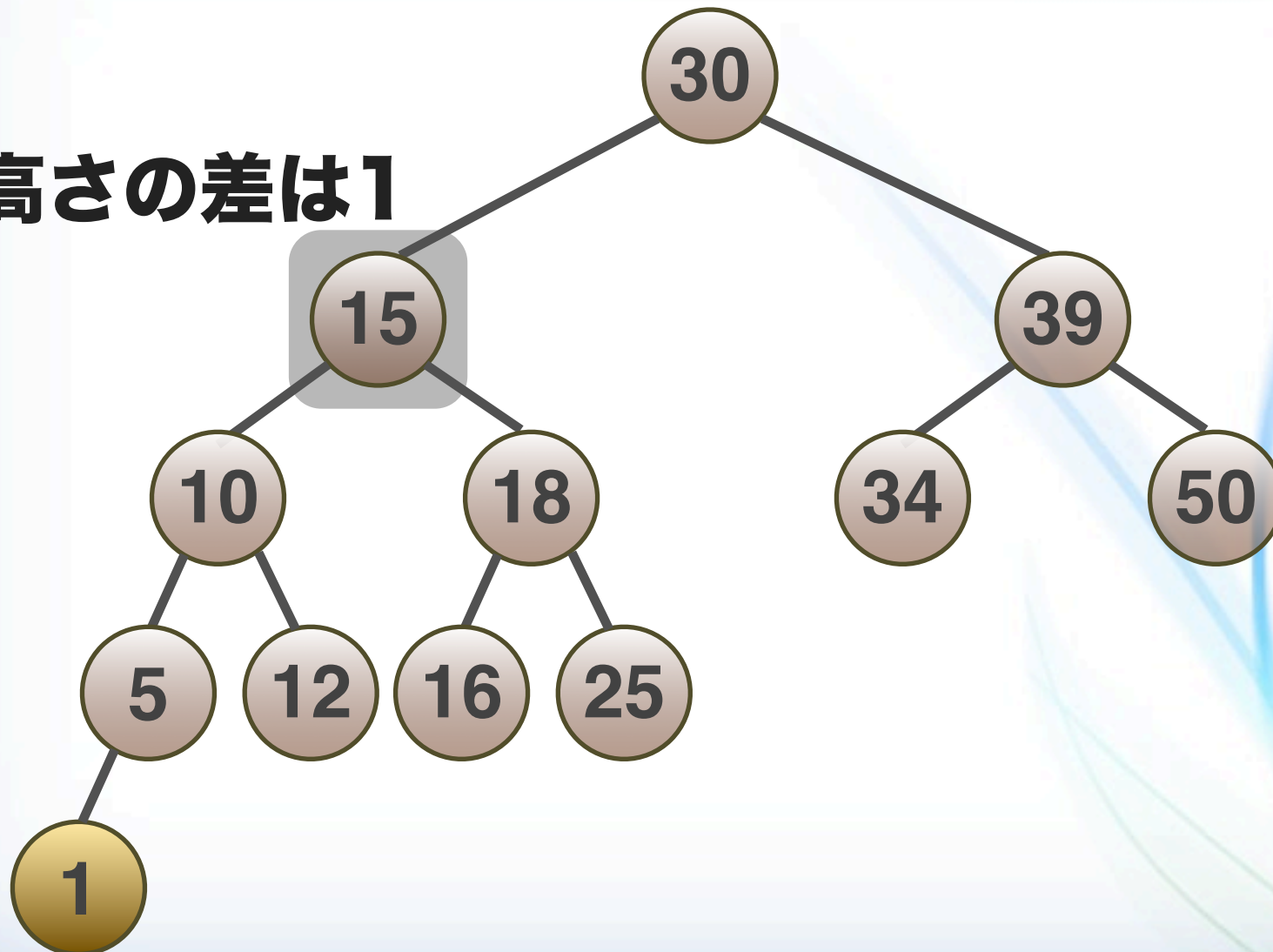
25



AVL木の事例 1

26

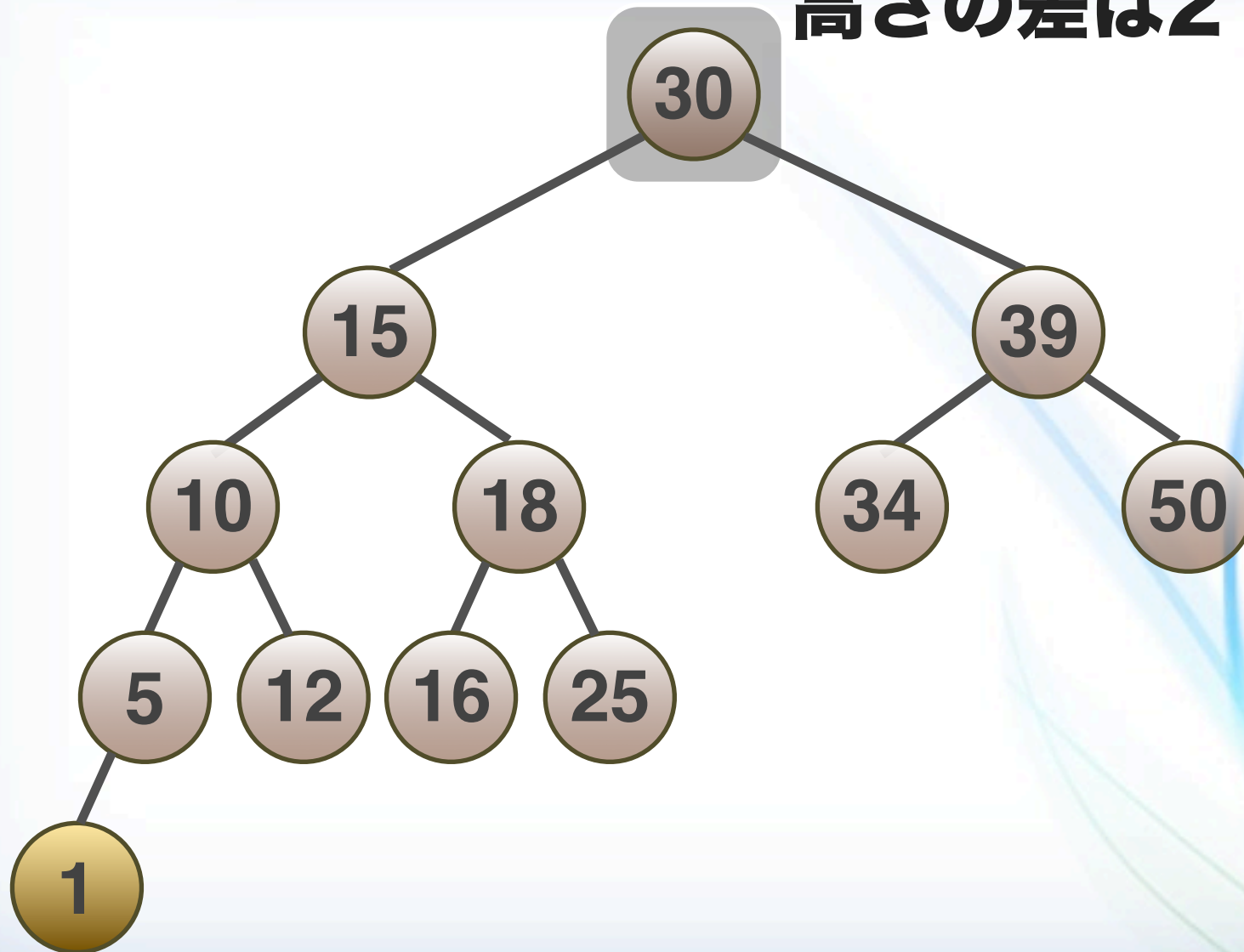
高さの差は1



AVL木の事例 1

27

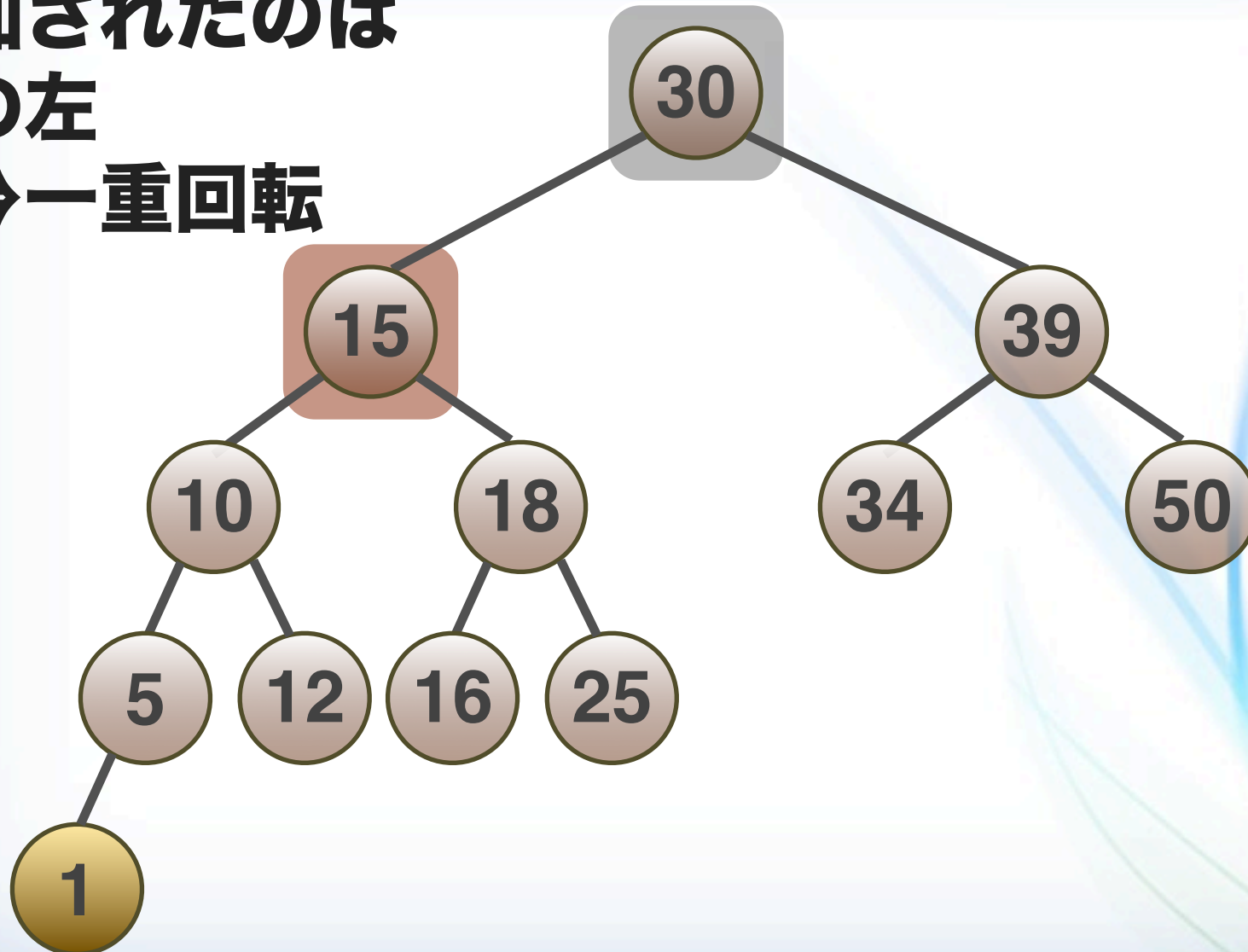
高さの差は2



AVL木の事例 1

28

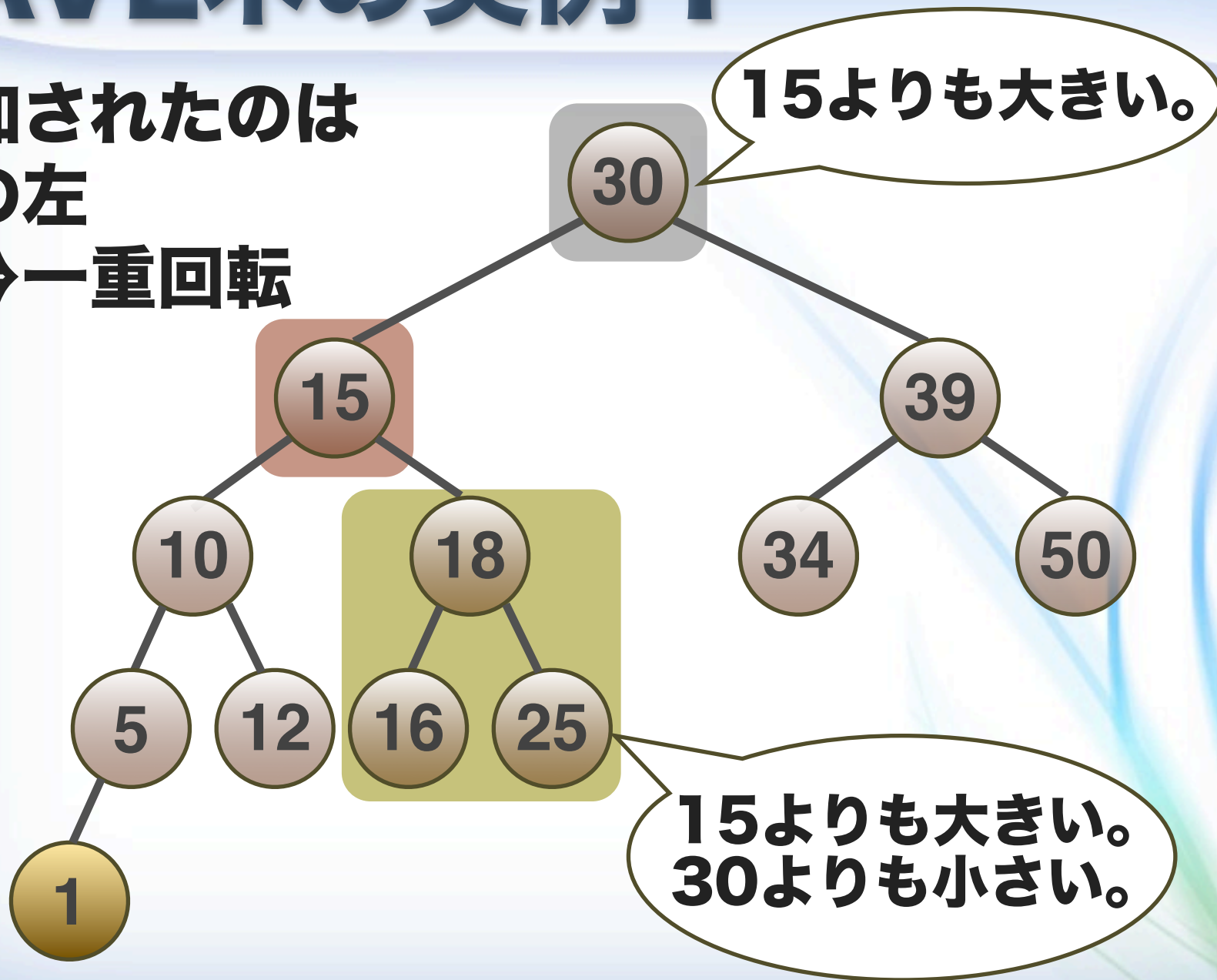
追加されたのは
左の左
⇒一重回転



AVL木の実例 1

29

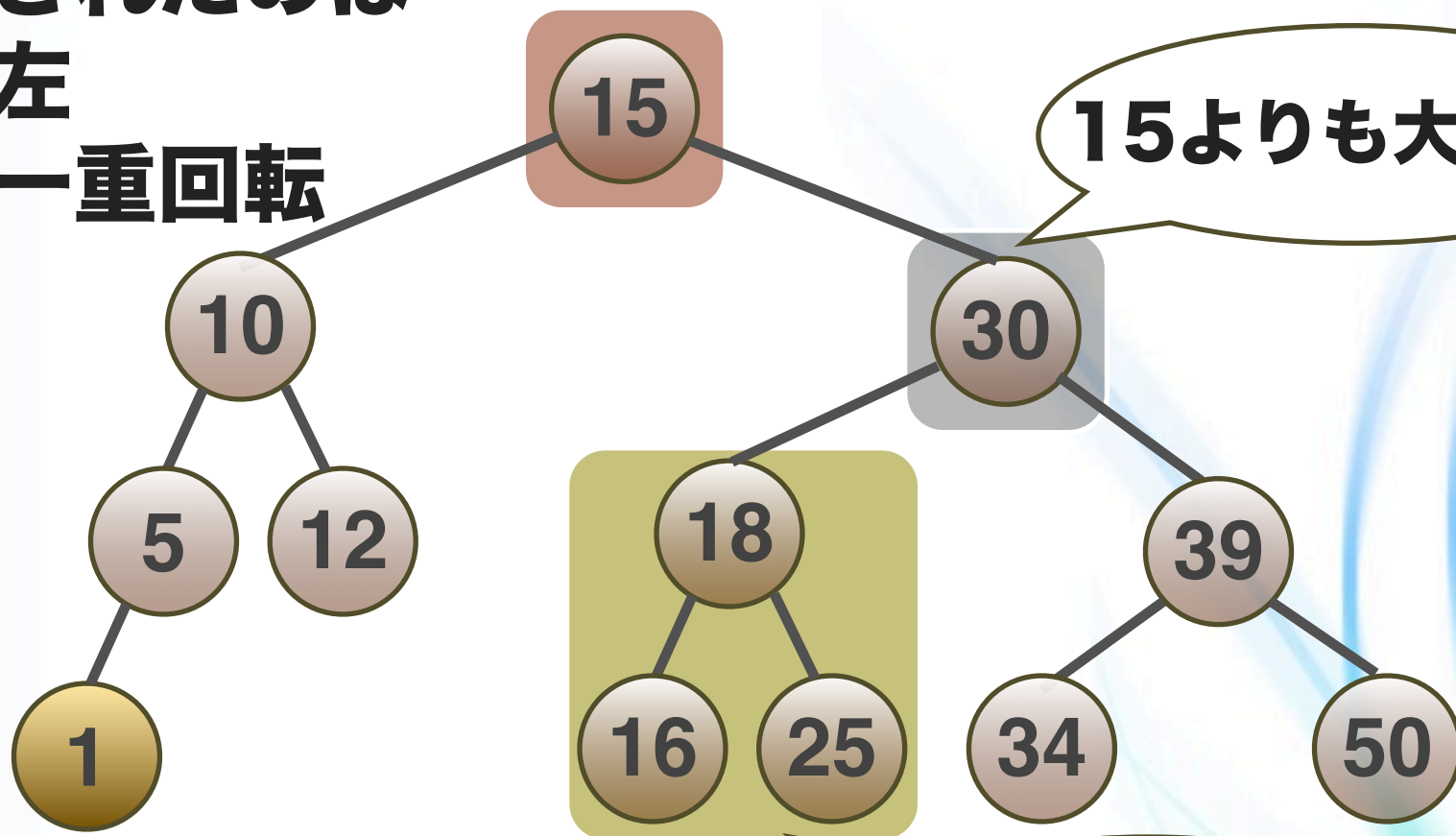
追加されたのは
左の左
⇒一重回転



AVL木の実例 1

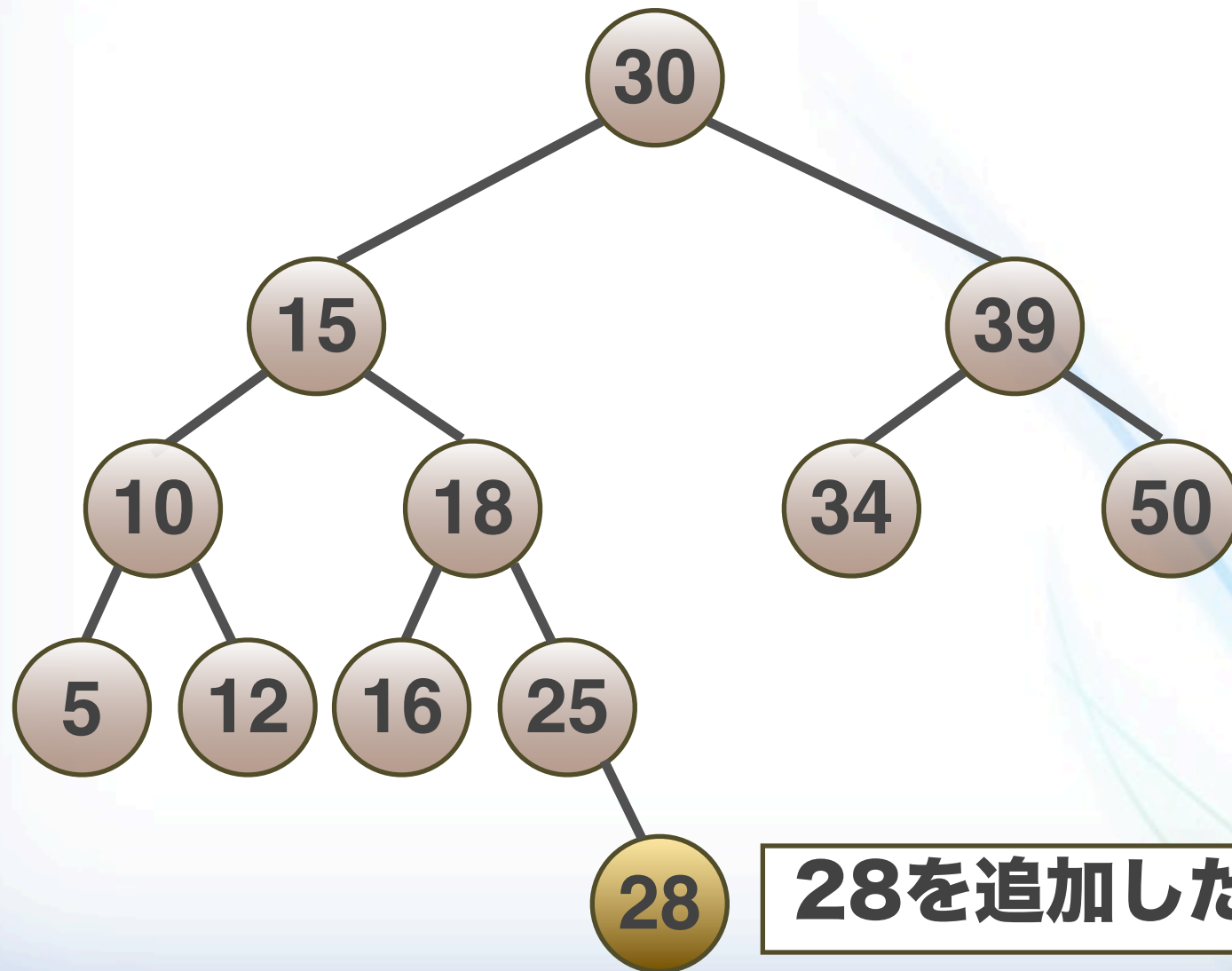
30

追加されたのは
左の左
⇒一重回転



AVL木の実例 2

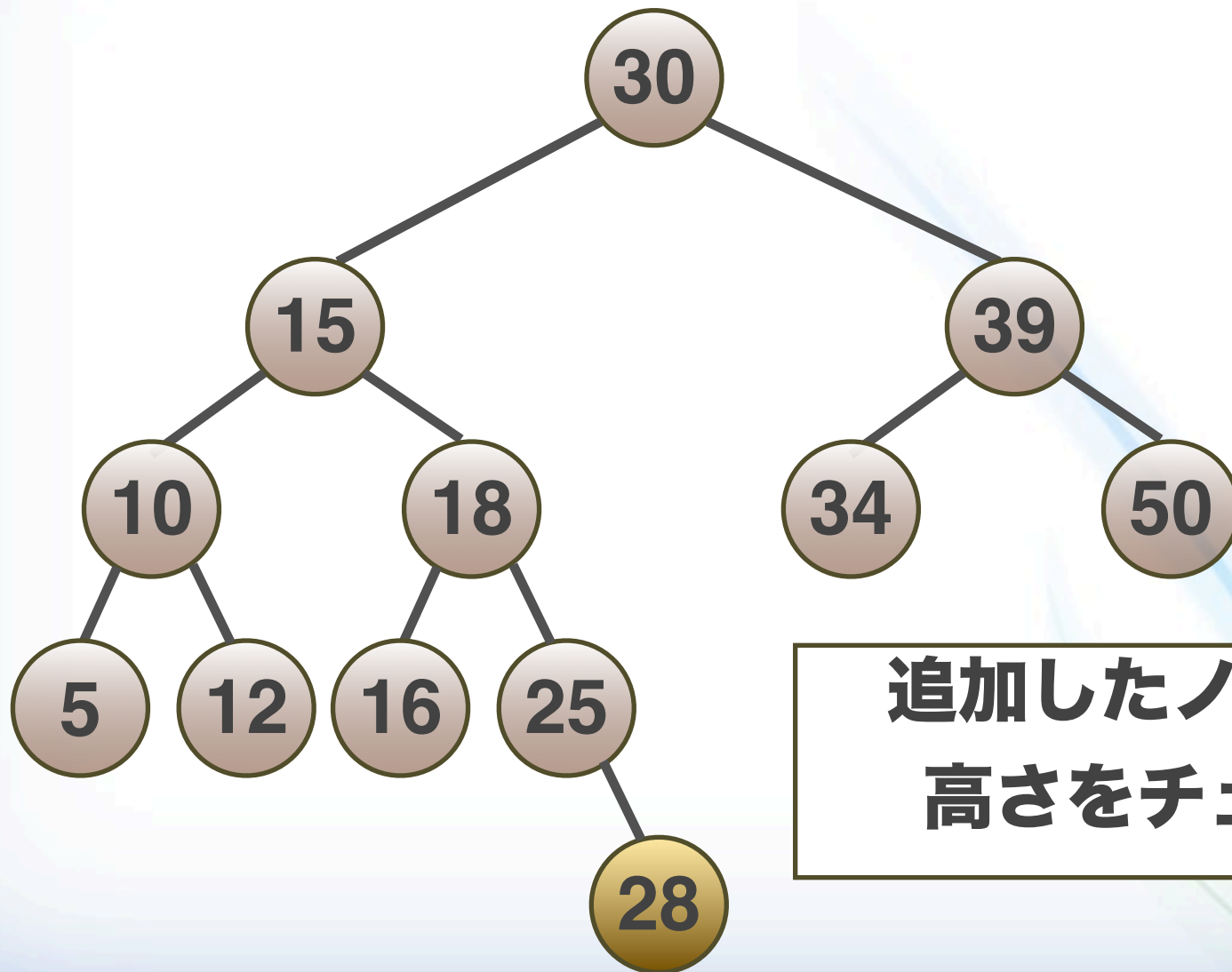
31



28を追加した。

AVL木の実例 2

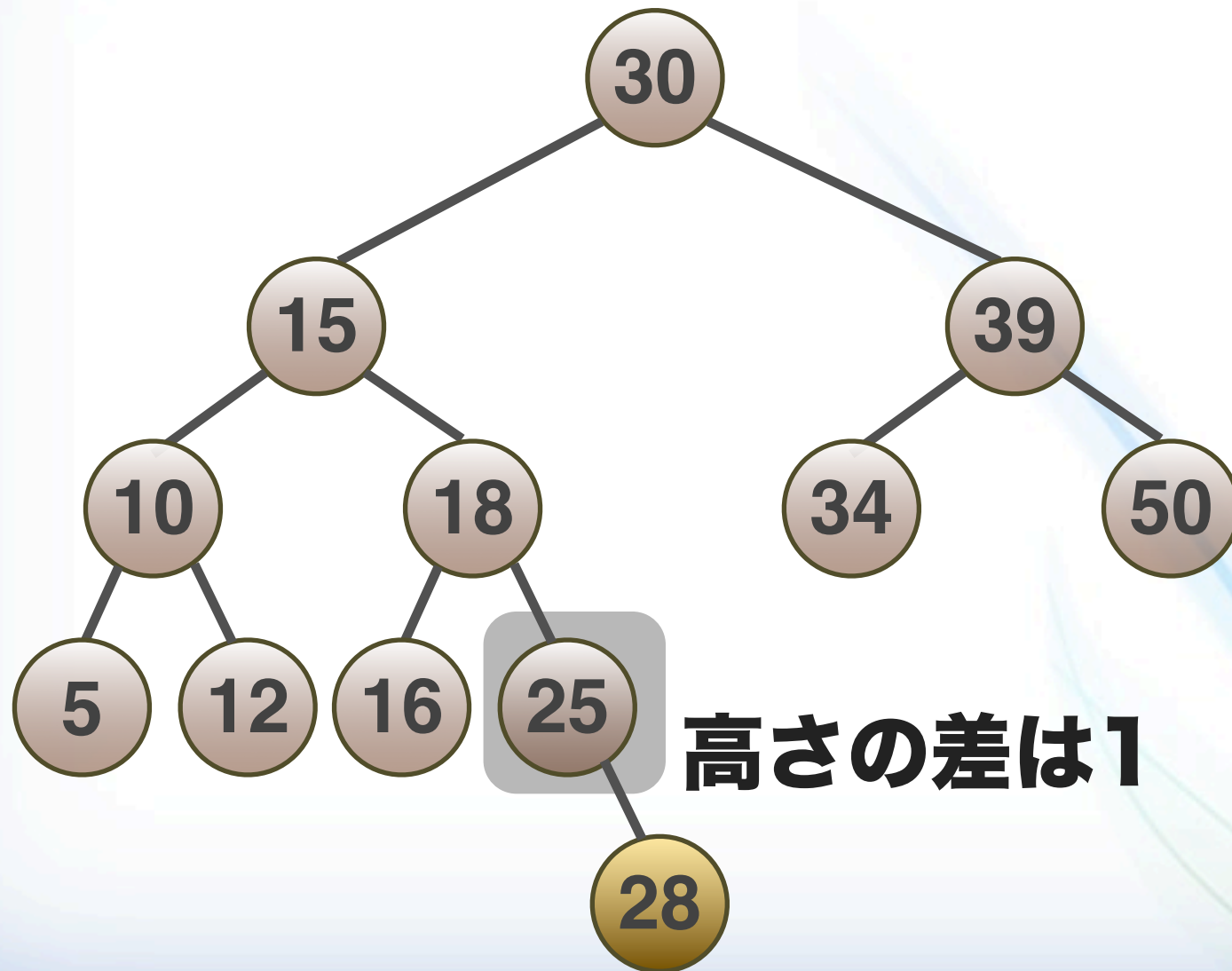
32



追加したノードから
高さをチェック。

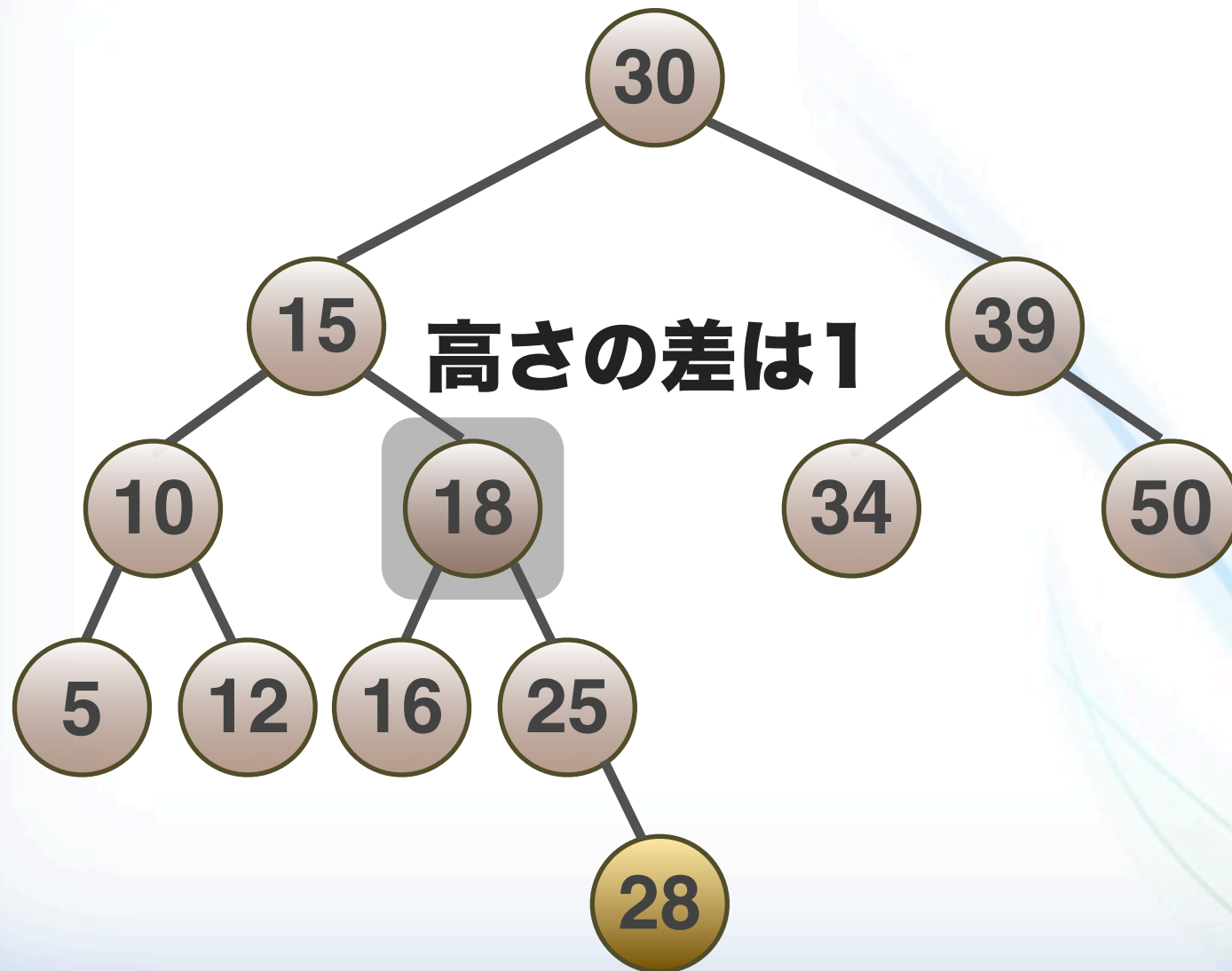
AVL木の実例 2

33



AVL木の事例 2

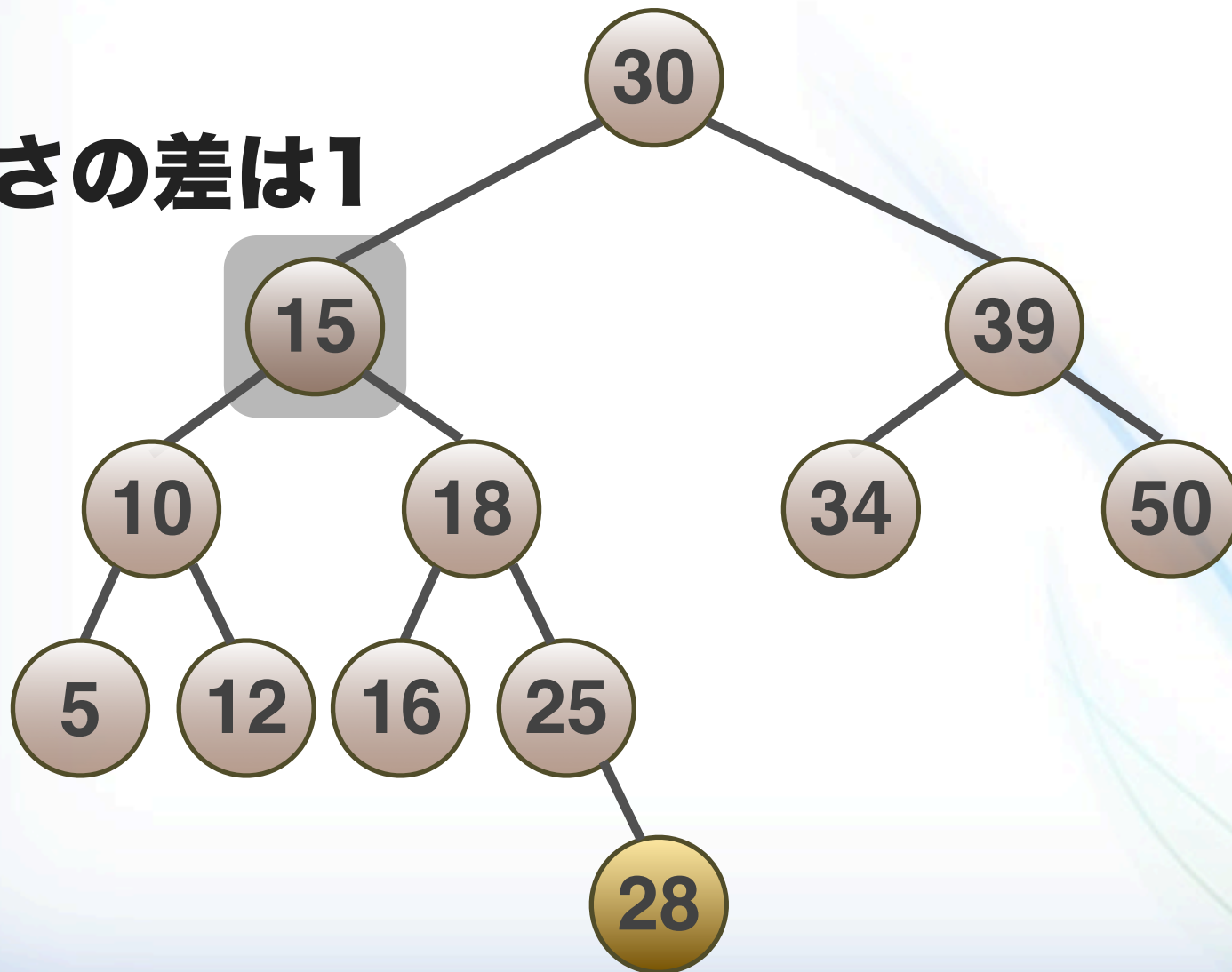
34



AVL木の実例 2

35

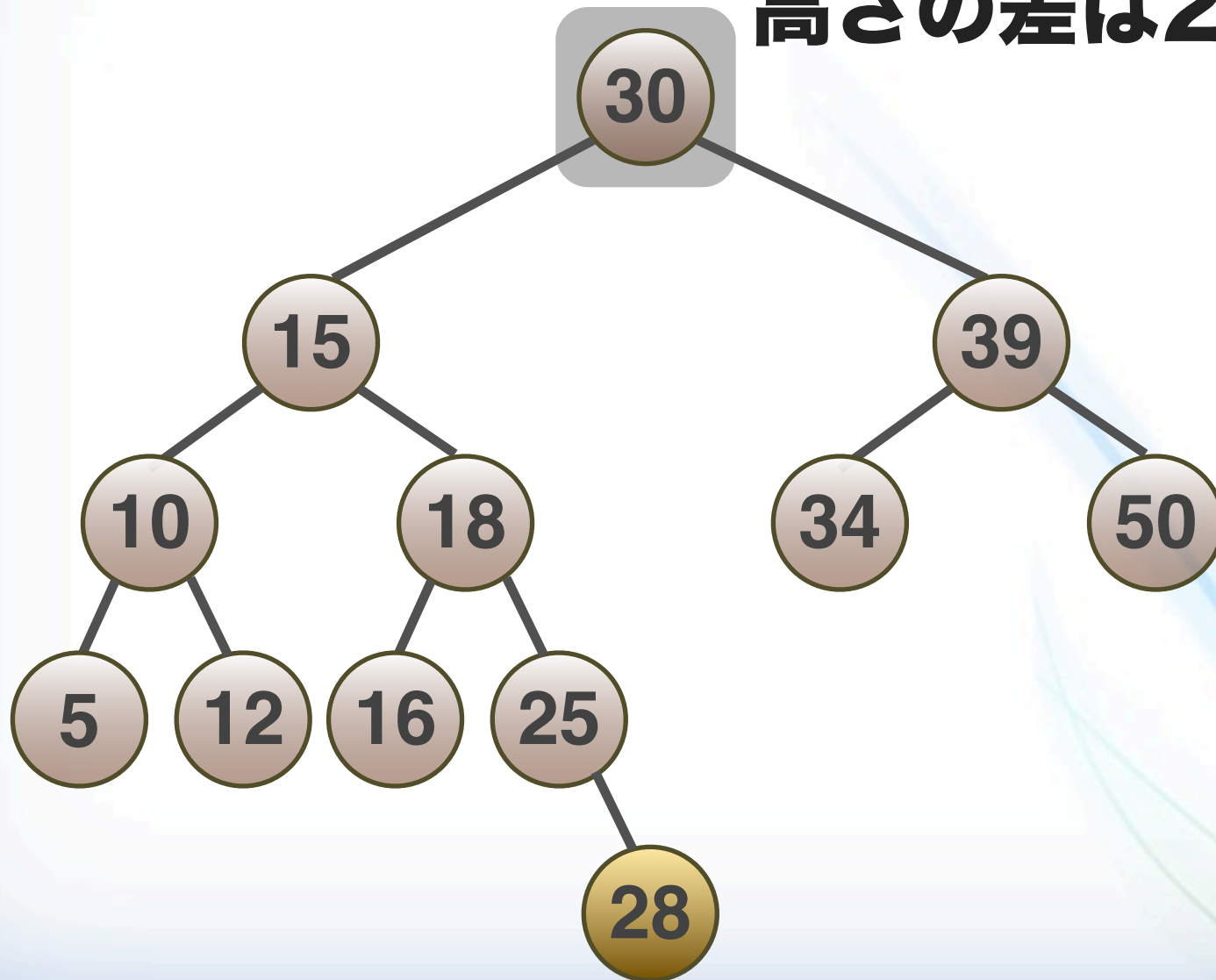
高さの差は1



AVL木の実例 2

36

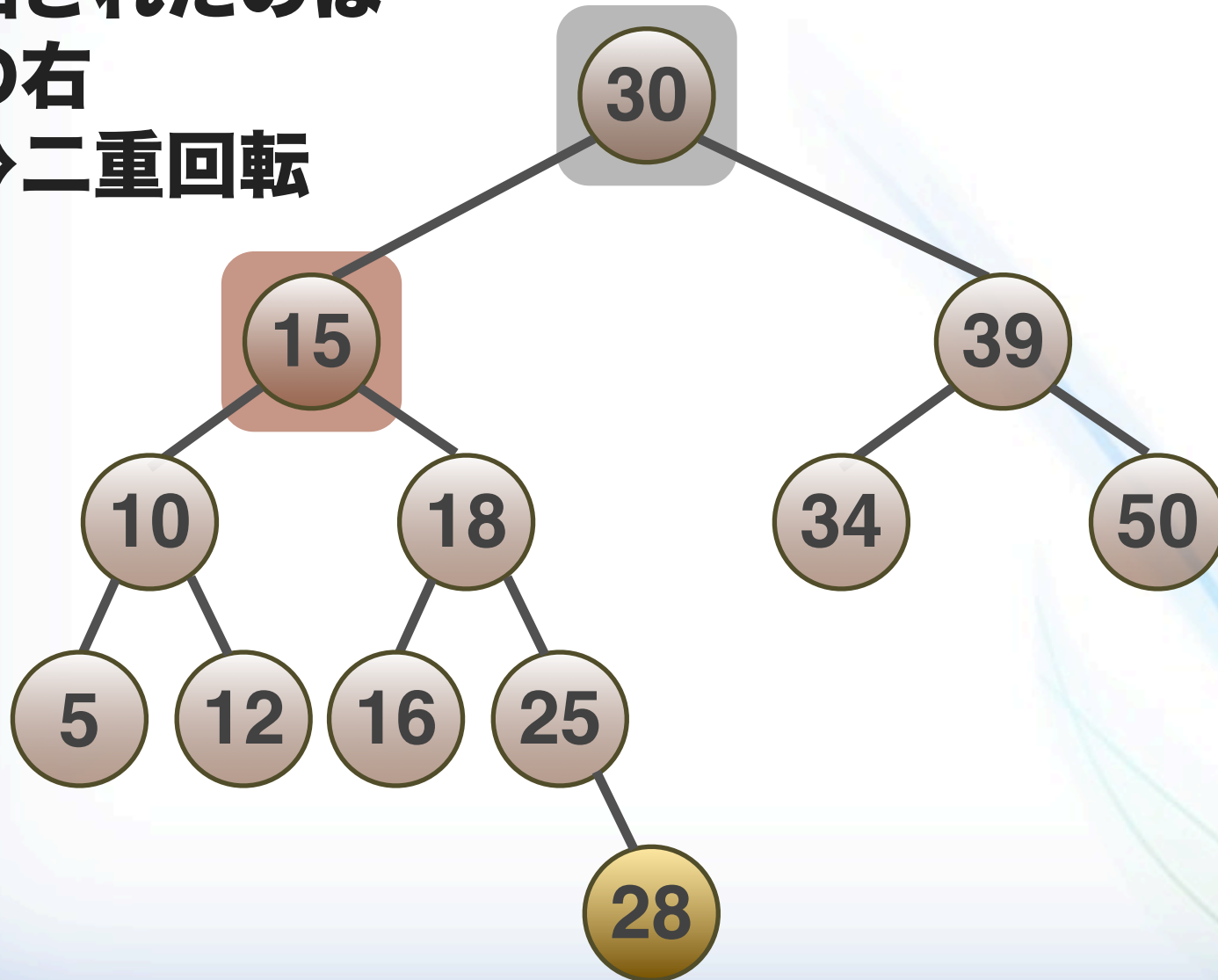
高さの差は2



AVL木の事例 2

37

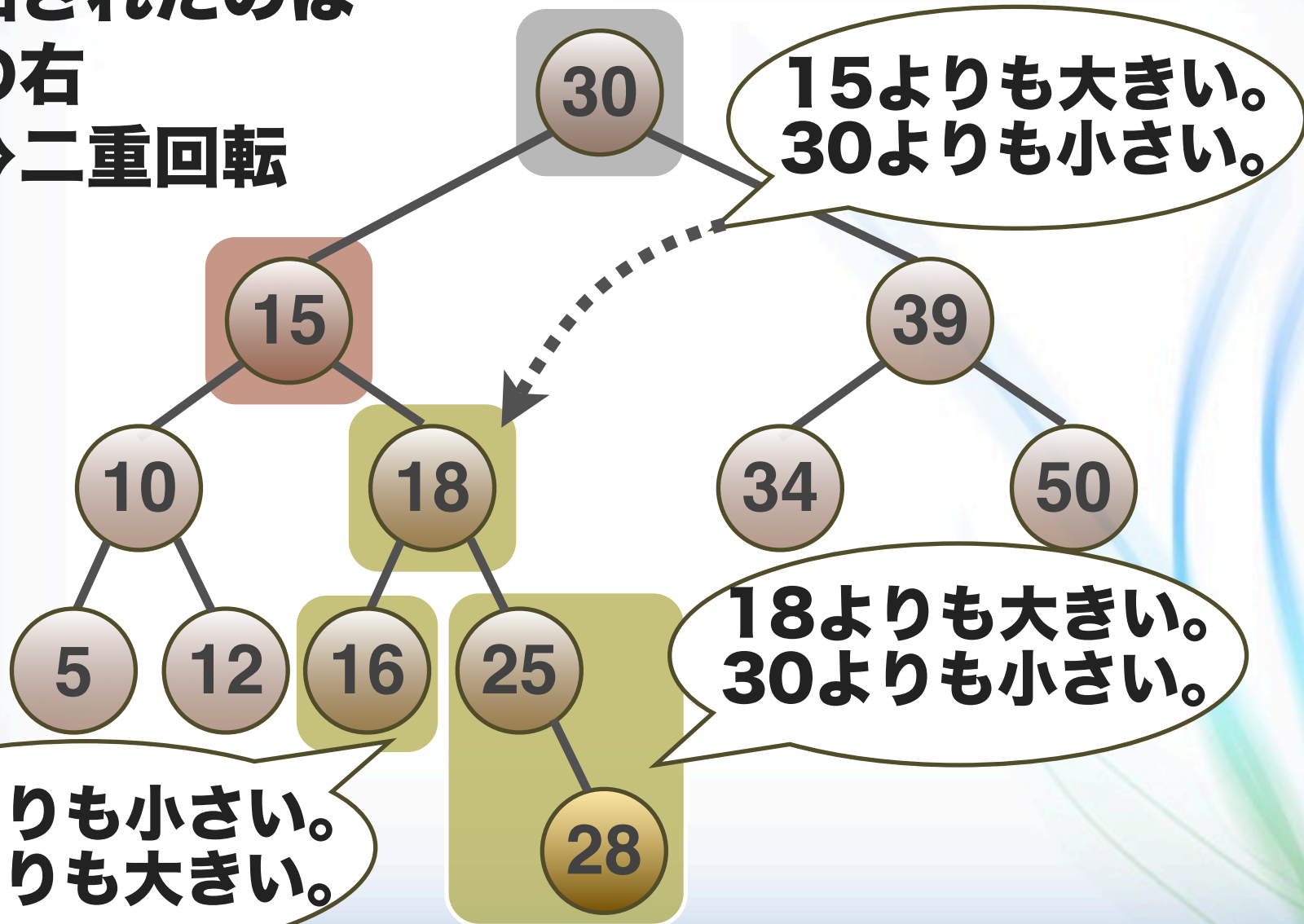
追加されたのは
左の右
⇒二重回転



AVL木の事例 2

38

追加されたのは
左の右
⇒二重回転

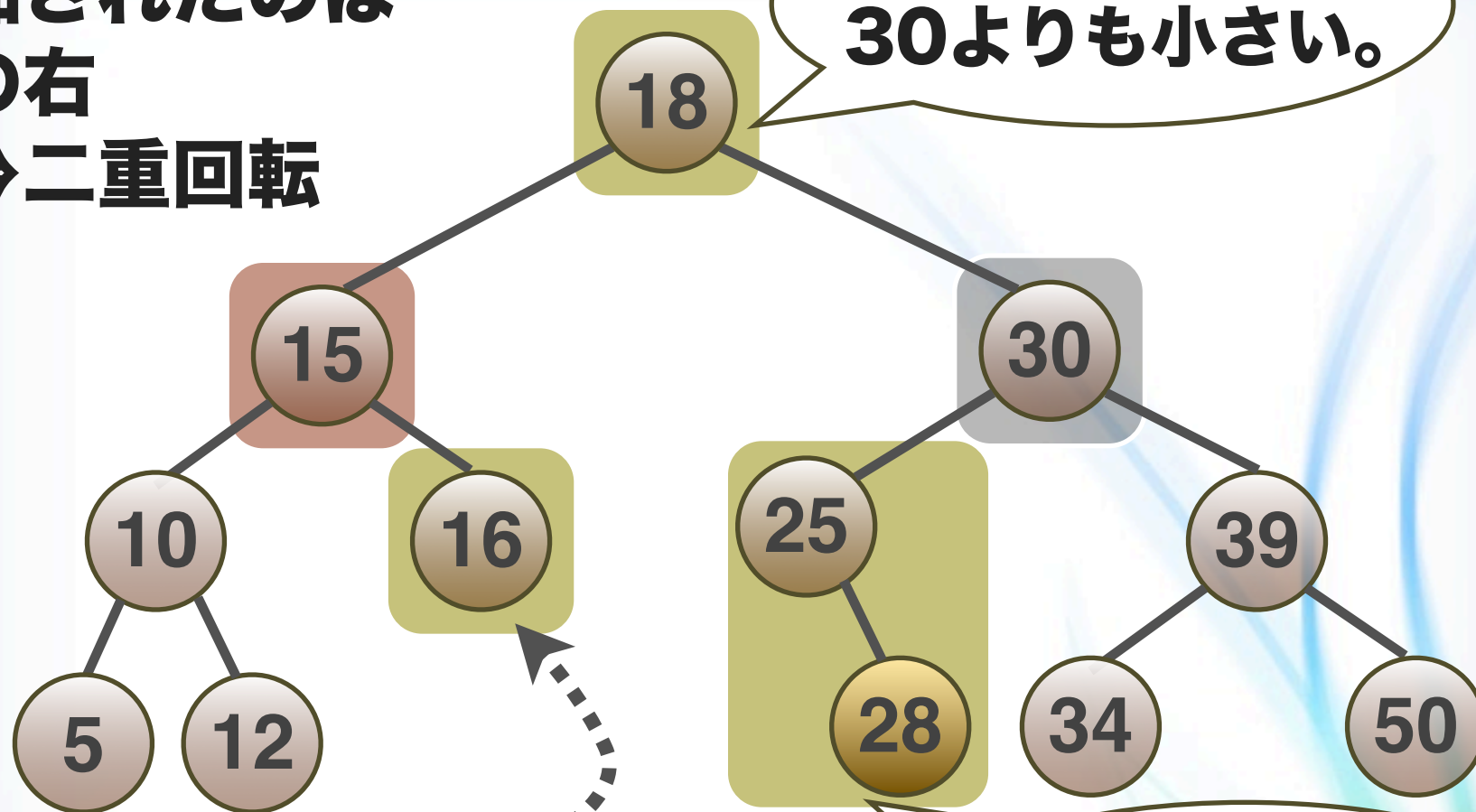


AVL木の事例 2

39

追加されたのは
左の右
⇒二重回転

15よりも大きい。
30よりも小さい。



18よりも小さい。
15よりも大きい。

18よりも大きい。
30よりも小さい。

AVL木の考察

40

- 「右の左」と「右の右」への追加は、左の場合の逆（線対称）。
- 削除する場合も追加と同様に高さをチェックする。
- 追加・削除のコストは、追加・削除の部分にまず $O(\lg n)$ 。整形のために高さを計算するところで、かなり大きめの $O(\lg n)$ 。整形自体は単純な操作なので $O(1)$ 。

$$O(\lg n) + O(\lg n) + O(1) = O(\lg n)$$

レポート課題

41

- 次のAVL木に45を追加する様子を書け。
- またその結果から77、92、30の順に削除する様子を書け。

