Exercise 1

Show that the longest simple path from a node x in a red-black tree to a descendant leaf has length at most twice that of the shortest simple path from node x to a descendant leaf.

Suppose we have the longest simple path (a_1; a_2; ⋯ ; a_s) and the shortest simple path (b_1; b_2; ⋯; b_t).

Suppose we have the longest simple path $(a_1; a_2; \cdots ; a_s)$ and the shortest simple path $(b_1; b_2; \cdots; b_t)$.

Then, by property 5 we know they have equal numbers of black nodes.

Suppose we have the longest simple path $(a_1; a_2; \cdots ; a_s)$ and the shortest simple path $(b_1; b_2; \cdots; b_t)$.

Then, by property 5 we know they have equal numbers of black nodes.

By property 4, we know that neither contains a repeated red node.

This tells us that at most $\left\lfloor \frac{s}{2} \right\rfloor$ of the nodes in the longest path are red.

Suppose we have the longest simple path (a_1; a_2; ⋯ ; a_s) and the shortest simple path (b_1; b_2; ⋯; b_t).

Then, by property 5 we know they have equal numbers of black nodes.

By property 4, we know that neither contains a repeated red node. This tells us that at most $\left\lfloor \frac{s}{2} \right\rfloor$ of the nodes in the longest path are red.

This means that at least $\left\lceil \frac{s}{2} \right\rceil$ are black, so, $\mathrm{t} \geq \left\lceil \frac{s}{2} \right\rceil$. So, if, by way of contradiction, we had that $s \geq 2\mathrm{t} + 1$, then $\mathrm{t} \geq \left\lceil \frac{s}{2} \right\rceil \geq \left\lceil \frac{2t+1}{2} \right\rceil = t + 1$, a contradiction.

Exercise 2

What is the largest possible number of internal nodes in a red-black tree with black-height k?

What is the smallest possible number?

In a path from root to leaf we can have at most one red node between any two black nodes, so maximal height of such a tree is 2k+1, where each path from root to leaf is alternating red and black nodes. To maximize internal nodes, we make the tree complete, giving a total of $2^{2k} - 1$ internal nodes.

In a path from root to leaf we can have at most one red node between any two black nodes, so maximal height of such a tree is 2k+1, where each path from root to leaf is alternating red and black nodes. To maximize internal nodes, we make the tree complete, giving a total of $2^{2k} - 1$ internal nodes.

The smallest possible number of internal nodes comes from a complete binary tree, where every node is black. This has $2^k - 1$internal nodes.

Exercise 3

An ***AVL tree*** is a binary search tree that is ***height balanced***: for each node x, the heights of the left and right subtrees of x differ by at most 1. To implement an AVL tree, we maintain an extra attribute in each node: x.*h* is the height of node x. As for any other binary search tree T , we assume that T.*root* points to the root node.

a. Prove that an AVL tree with n nodes has height O(lg n). (*Hint:* Prove that an AVL tree of height h has at least F_h nodes, where F_h is the *h* th Fibonacci number.)

Let T(h) denote the minimum size of an AVL tree of height h. Since it is height h, it must have the max of it's children's heights is equal to h-1. Because of the restriction of AVL trees, we have that the smaller child must be at least one less than the larger one, so, we have that

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

$$\mathrm{T(h)} + 1 = [T(h-1) + 1] + [\mathrm{T(h} - 2) + 1]$$

$$F_h = F_{h-1} + F_{h-2}$$

黄金分割率及其共轭数是方程$x^2 = x + 1$的两个根

$$\Phi = \frac{1+\sqrt{5}}{2} = 1.618\ldots$$

$$\hat{\Phi} = \frac{1-\sqrt{5}}{2} = -0.618\ldots$$

斐波那契数与黄金分割率有关

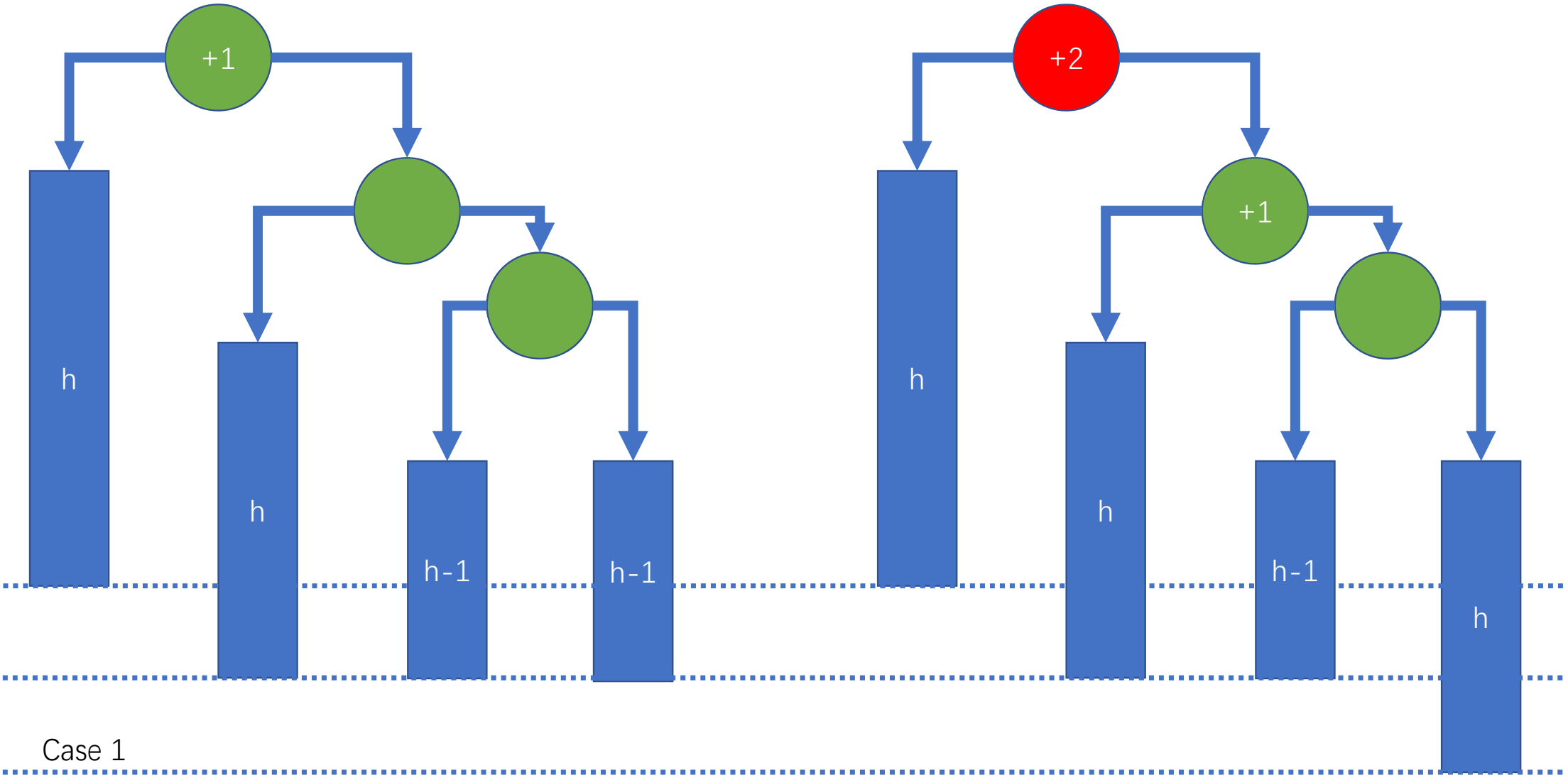
$$F_i = \frac{\Phi^i - \hat{\Phi}^i}{\sqrt{5}}$$

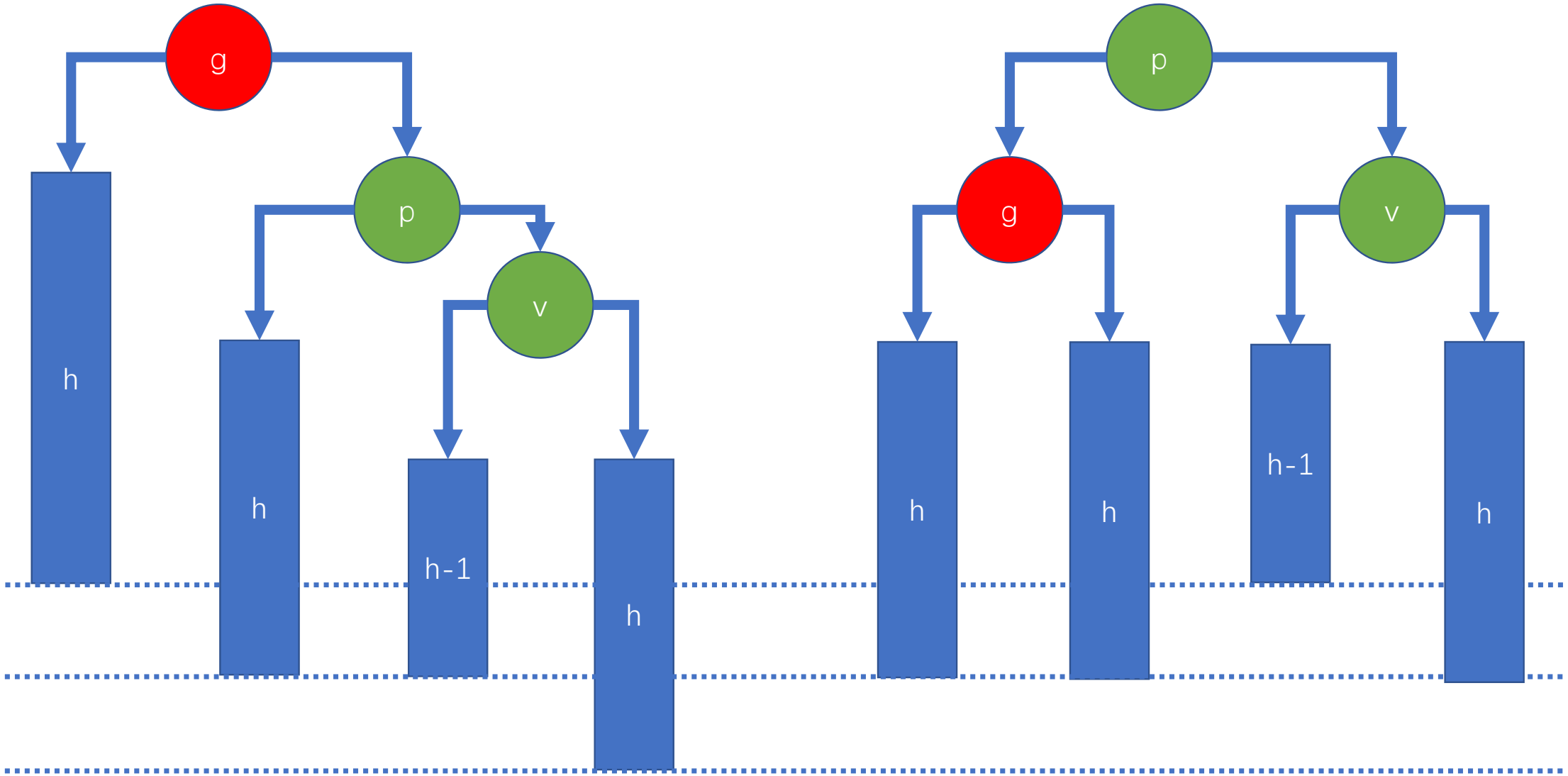
$$\frac{|\hat{\Phi}^i|}{\sqrt{5}} < \frac{1}{\sqrt{5}} < \frac{1}{2}$$

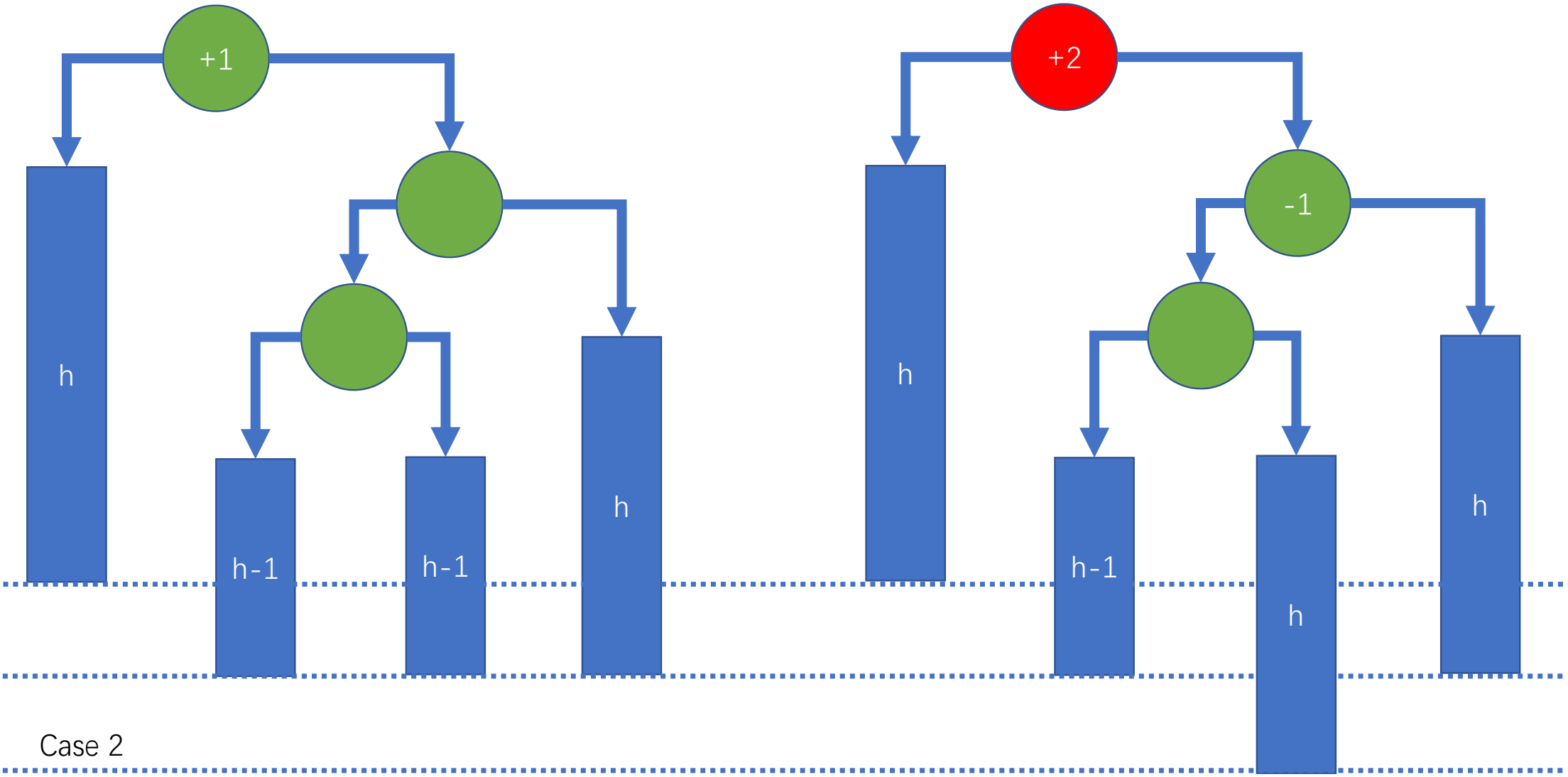
$$F_i = \left\lfloor \frac{\Phi^i}{\sqrt{5}} + \frac{1}{2} \right\rfloor$$

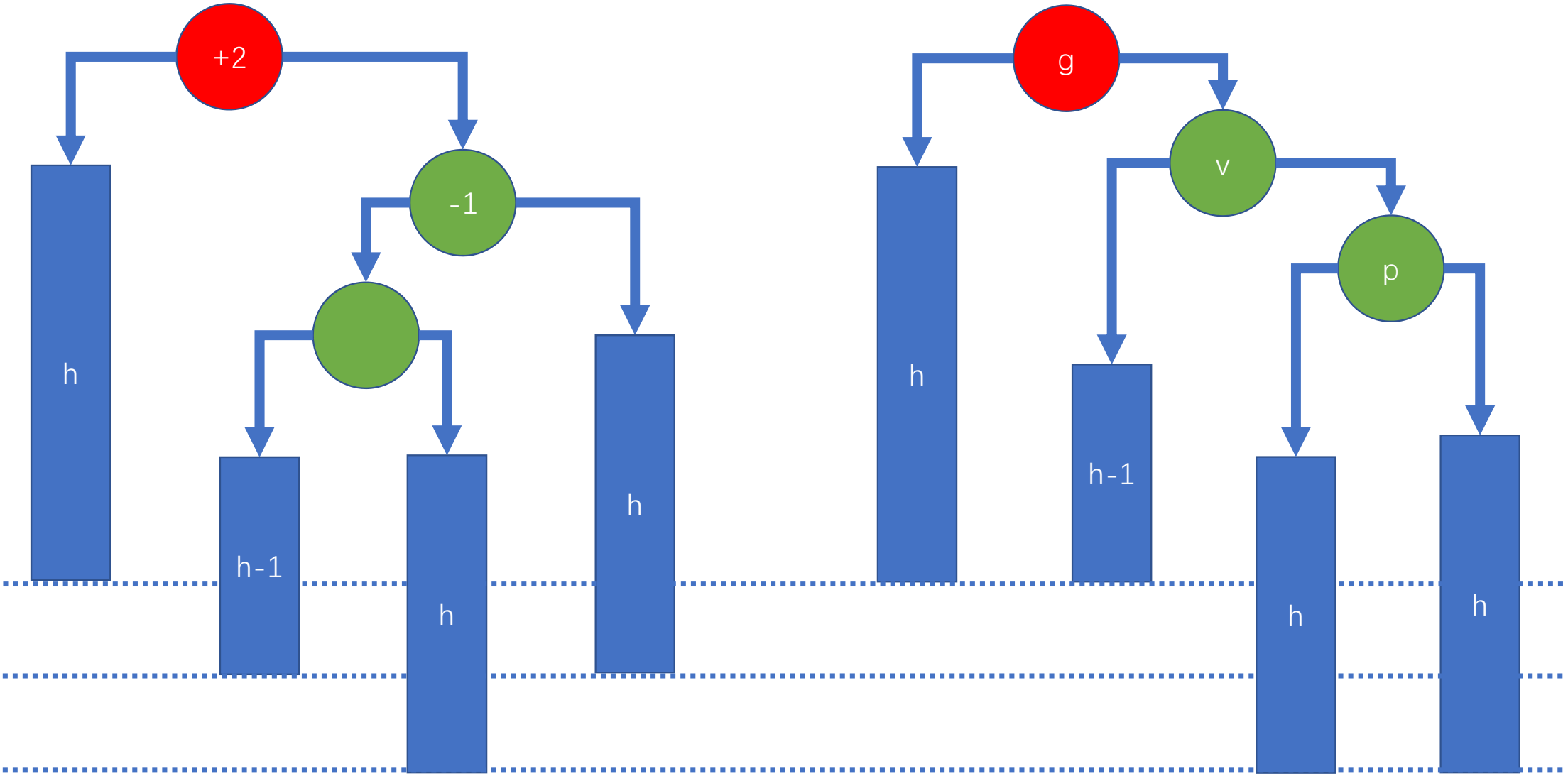
因此，斐波那契数以指数形式增长，所以树高O(lg n)

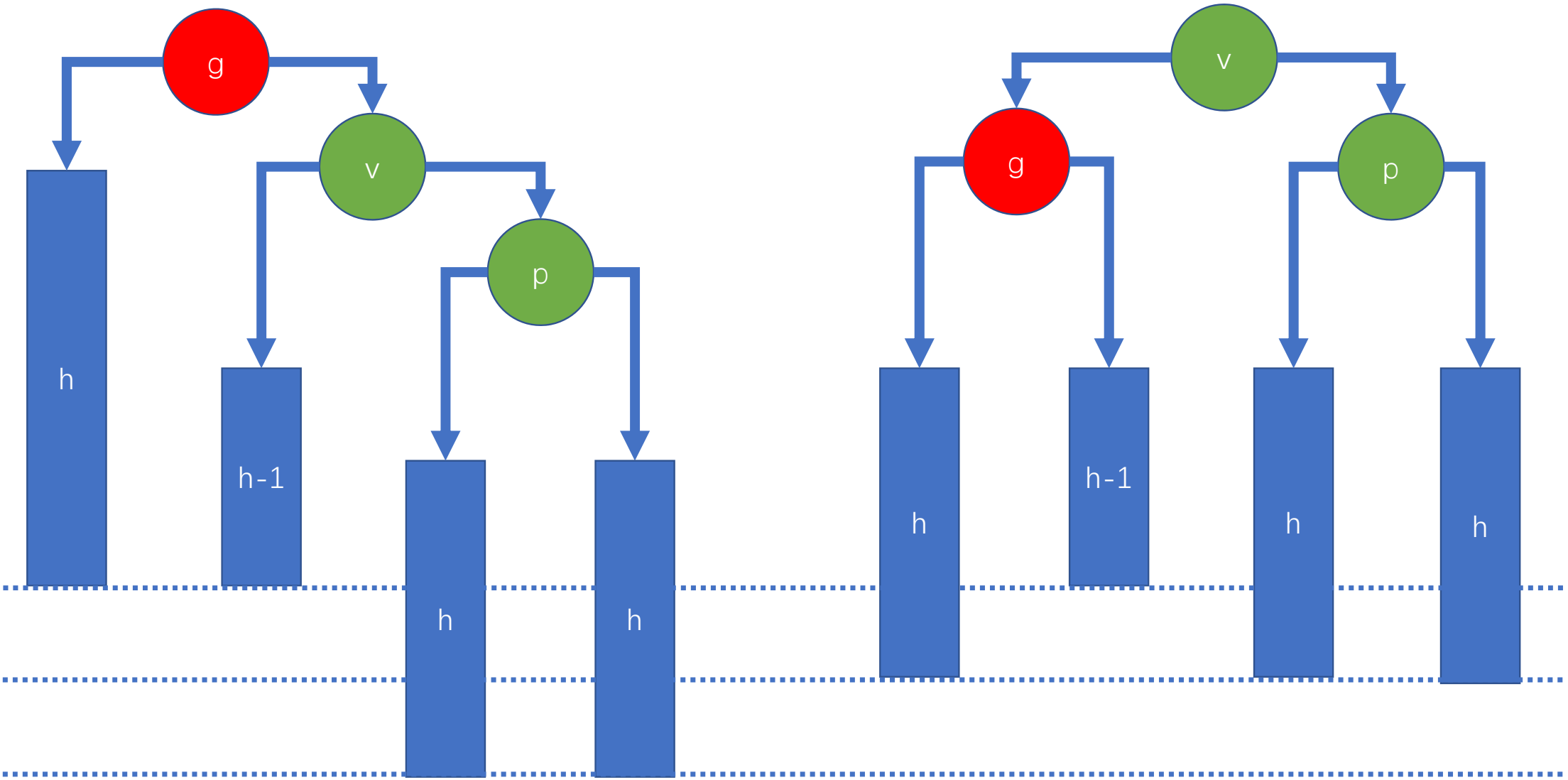
b. To insert into an AVL tree, we first place a node into the appropriate place in binary search tree order. Afterward, the tree might no longer be height balanced. Specifically, the heights of the left and right children of some node might differ by 2. Describe a procedure BALANCE(x), which takes a subtree rooted at x whose left and right children are height balanced and have heights that differ by at most 2, and alters the subtree rooted at x to be height balanced. (*Hint:* Use rotations.)

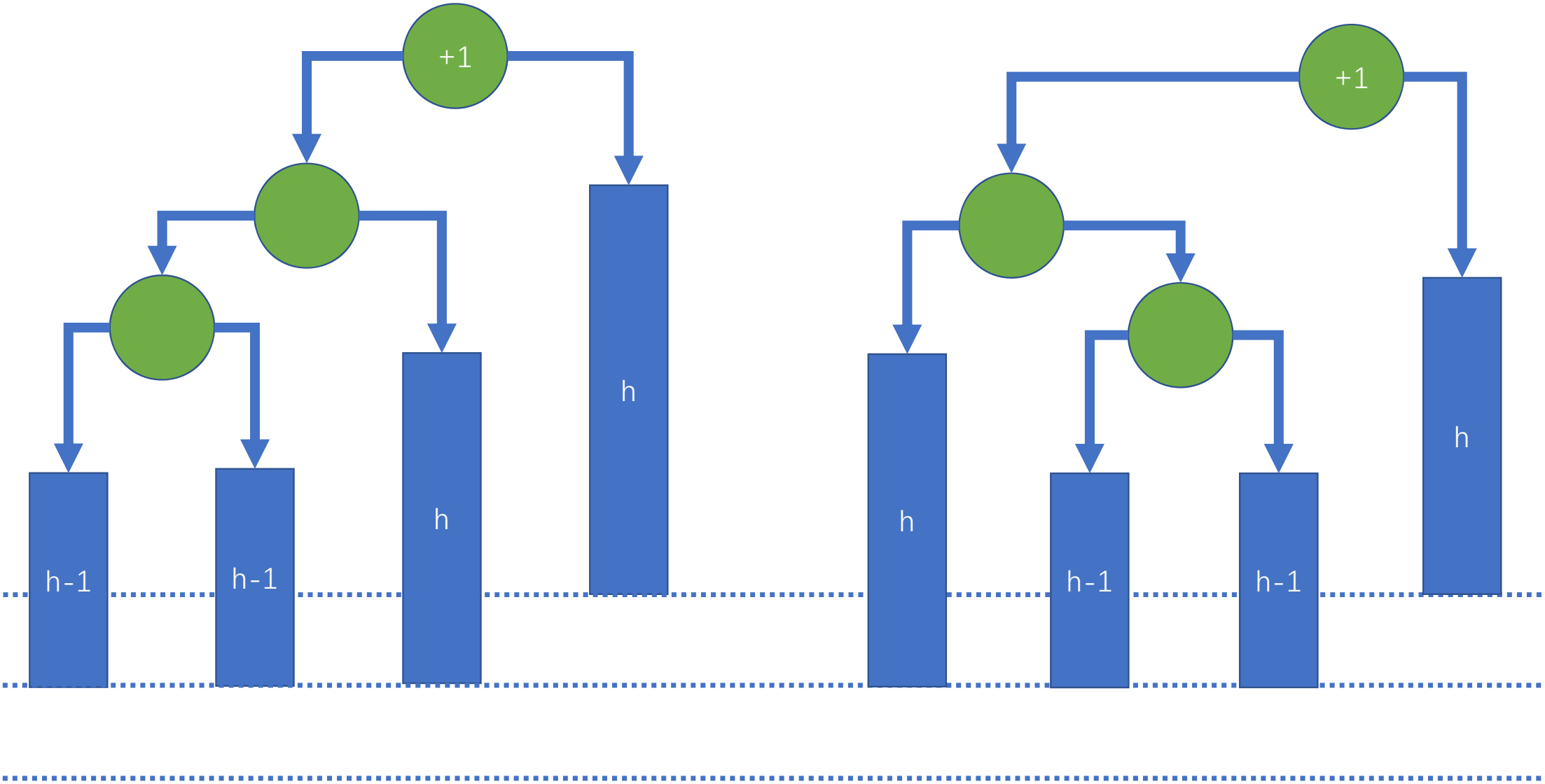

+1
h
h
h-1
h-1
+2
h
+1
h
h-1
h
Case 1


g
p
v
h
h
h-1
h
p
g
v
h
h
h-1
h


+1
h
h-1
h-1
h
+2
h
-1
h-1
h
h
Case 2


+2
h
-1
h-1
h
h
g
h
v
h-1
p
h
h


g
v
p
h
h-1
h
h
v
g
p
h
h-1
h
h


+1
h-1
h-1
h
h
+1
h
h-1
h-1
h