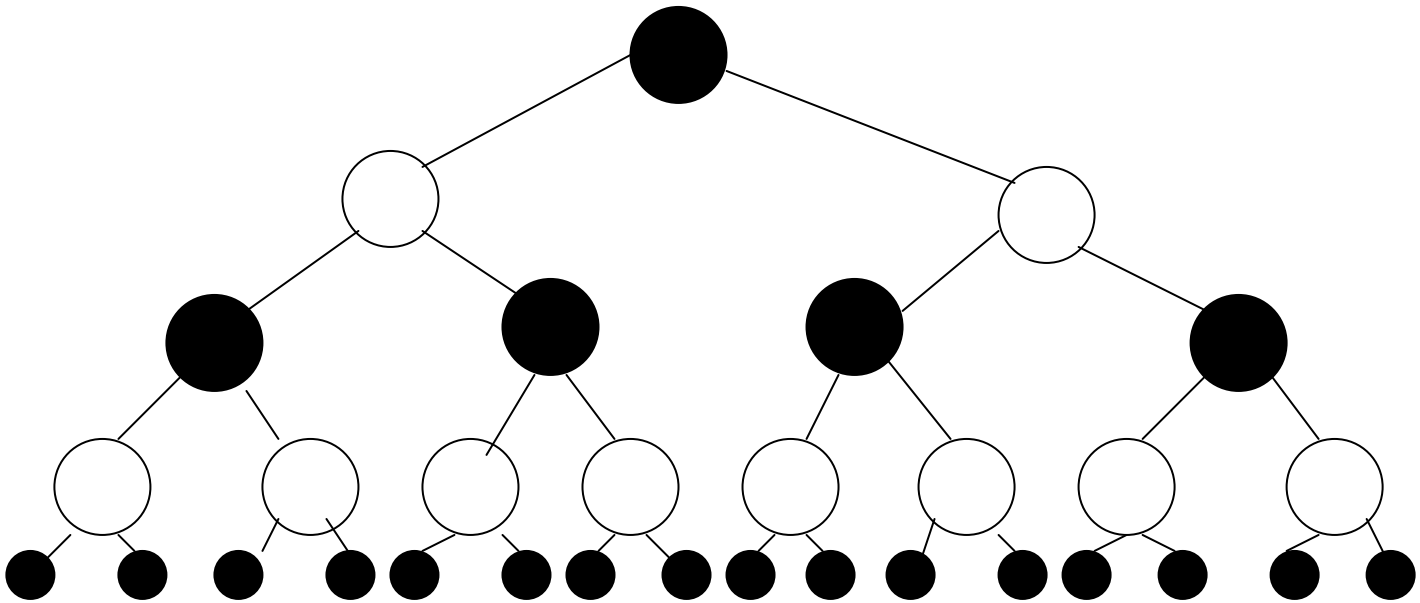
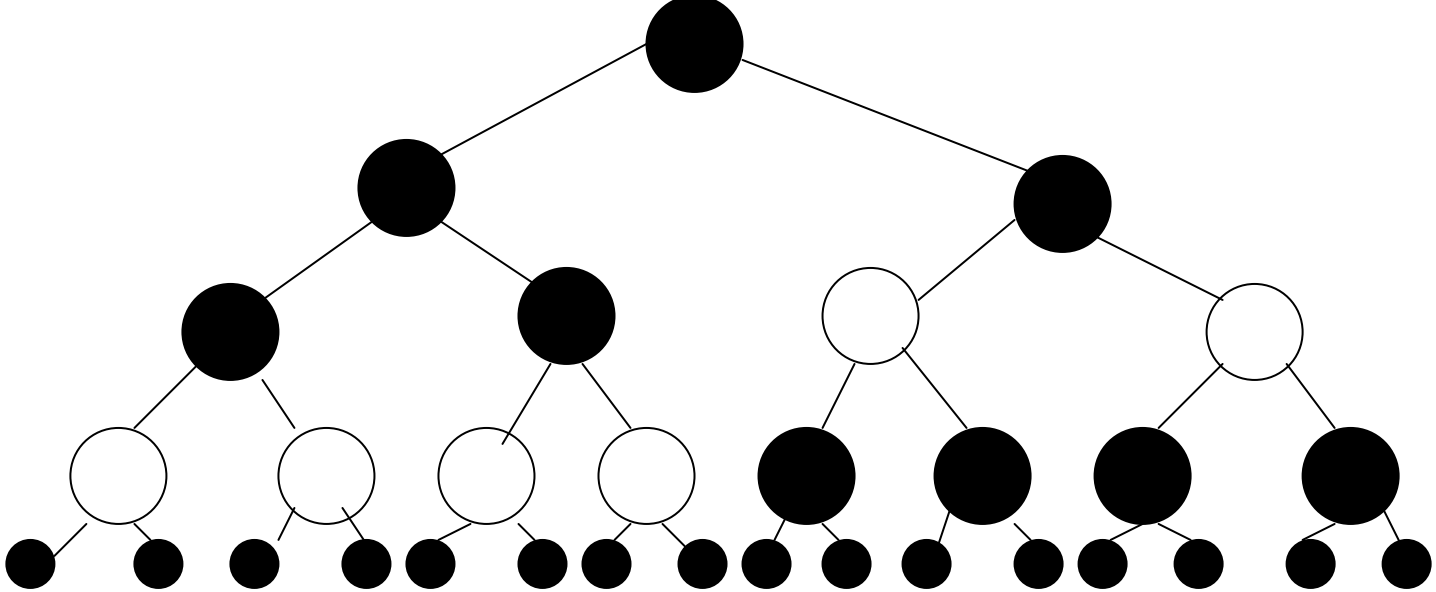
Lecture 2: Cây đỏ đen (Red-Black Trees)

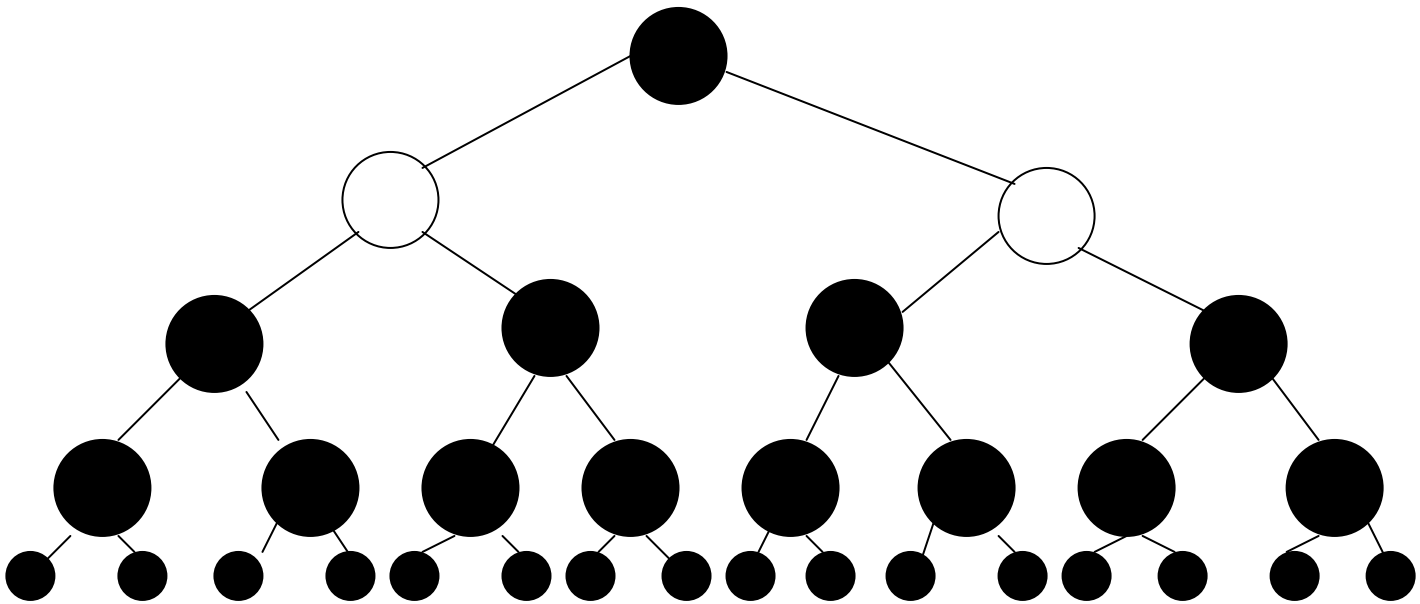
- Định nghĩa: Là cây NPTK, trong đó:
  - Nút gốc phải là nút đen
  - Mỗi nút hoặc là đỏ hoặc đen.
  - Mỗi con trỏ null được coi là một nút đen.
  - Một nút đỏ thì 2 nút con của nó phải là đen.
  - Mọi đường dẫn từ một nút đến một nút lá phải có cùng số nút đen.
- Định nghĩa: Chiều cao đen của một nút P bằng số nút đen trên đường dẫn từ nút đó đến một nút lá bất kỳ (không tính nút con trỏ NULL).



A valid Red-Black Tree

Black-Height = 2





Định lý 1: Một cây đỏ đen bất kỳ có gốc là nút x, có ít nhất n = $2^{bh(x)} - 1$ nút trong với bh(x) chiều cao đen của nút gốc x.

Chứng minh bằng quy nạp trên chiều cao của x.

- Nếu bh=0, thì nút gốc x=NULL, như vậy số nút trong bằng 0 = $\mathbf{2^{bh(x)} - 1}$ =0
- Nếu bh=1, thì nút gốc x<> NULL, số nút trong ít nhất là 1= $\mathbf{2^{bh(x)} - 1}$ =1
- Giả sử cây có chiều cao đen là bh(x) => số nút trong của cây này bằng tổng số nút trong của 2 cây con cộng với 1: Các cây con của x có thể có chiều cao là bh(x) nếu như nút x là đỏ, bh(x)-1 nếu như x là đen. Như vậy số nút trong của cây gốc x ít nhất là:

$$\mathbf{(2^{bh(x)-1} - 1) + (2^{bh(x)-1} - 1) + 1 = (2^{bh(x)} - 1)}$$

Định lý 2: Trong một cây đỏ đen, ít nhất một nửa số nút trên đường dẫn từ gốc đến lá là nút đen.

Chứng minh: dựa theo tính chất một nút đỏ thì 2 nút con của nó phải đen.

Định lý 3 –Một cây đỏ-đen có n nút trong có chiều cao h <= 2 lg(n + 1).

Chứng minh: gọi h là chiều cao của cây đỏ-đen có gốc x.

Theo địng lý 2:

$$bh(x) >= h/2$$

Từ định lý 1, ta có $n >= 2^{bh(x)} - 1$

$$=> \quad n >= 2^{h/2} - 1$$

$$n + 1 >= 2^{h/2}$$

$$\lg(n + 1) >= h/2$$

$$2\lg(n + 1) >= h$$

Bottom –Up Insertion

- Việc thêm nút được thực hiện như trong cây NPTK
- Màu của nút thêm vào ban đầu là màu RED
- Kiểm tra xem tính chất RB có bị vi phạm không?
  - Mỗi nút hoặc là Red hoặc Black
  - Con trỏ NULL được xem là một nút đen
  - Một nút đỏ thì 2 nút con của nó phải là đen
  - Mọi nút phải có cùng chiều cao đen
  - Nút gốc phải mầu đen

Bottom Up Insertion

- Thêm nút: màu ban đầu của nút là đỏ, X trỏ đến nút vừa thêm
- Cases

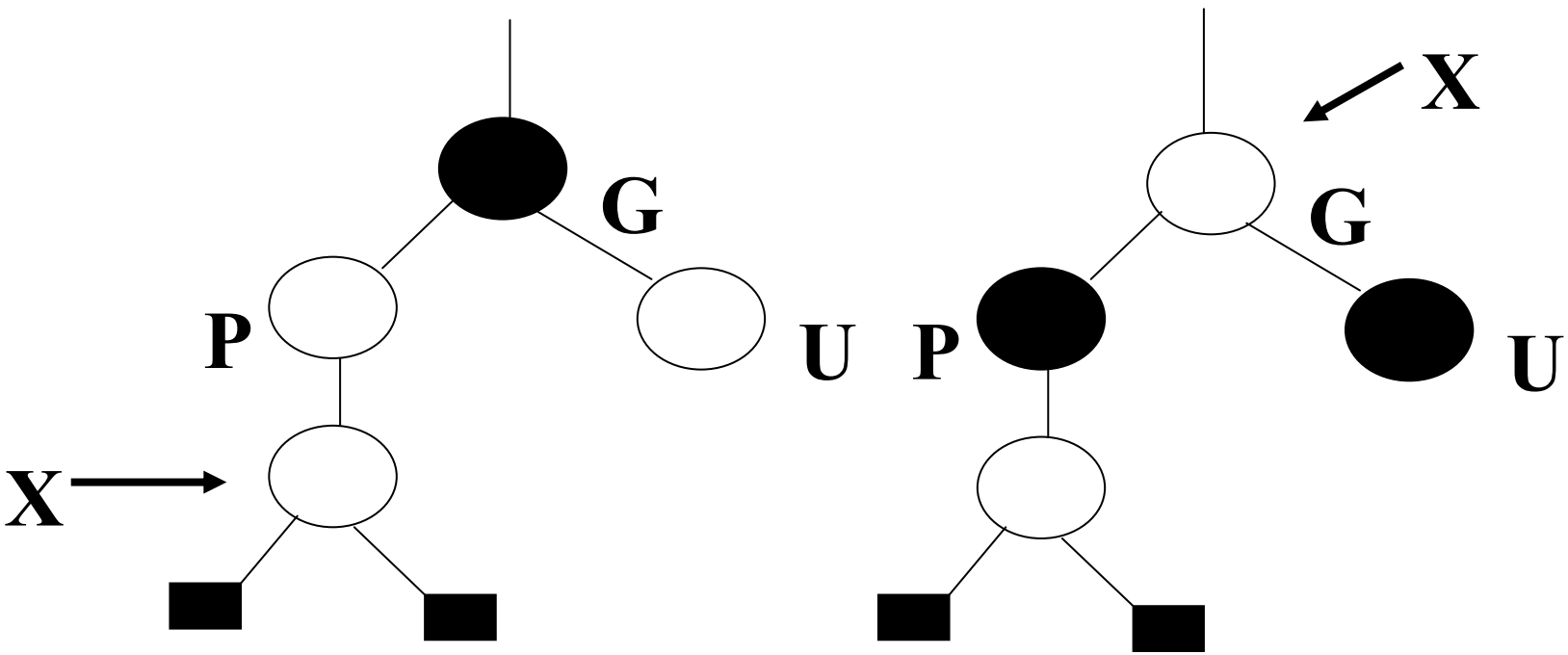
  0: X là nút gốc -> đổi mầu x thành đen.

  1: Cả 2 nút: cha và chú đều đỏ thì thực hiện đổi mầu nút cha, chú thành đen, màu nút ông thành đỏ, X trỏ đến nút ông, tiếp tục kiểm tra X tại vị trí mới.

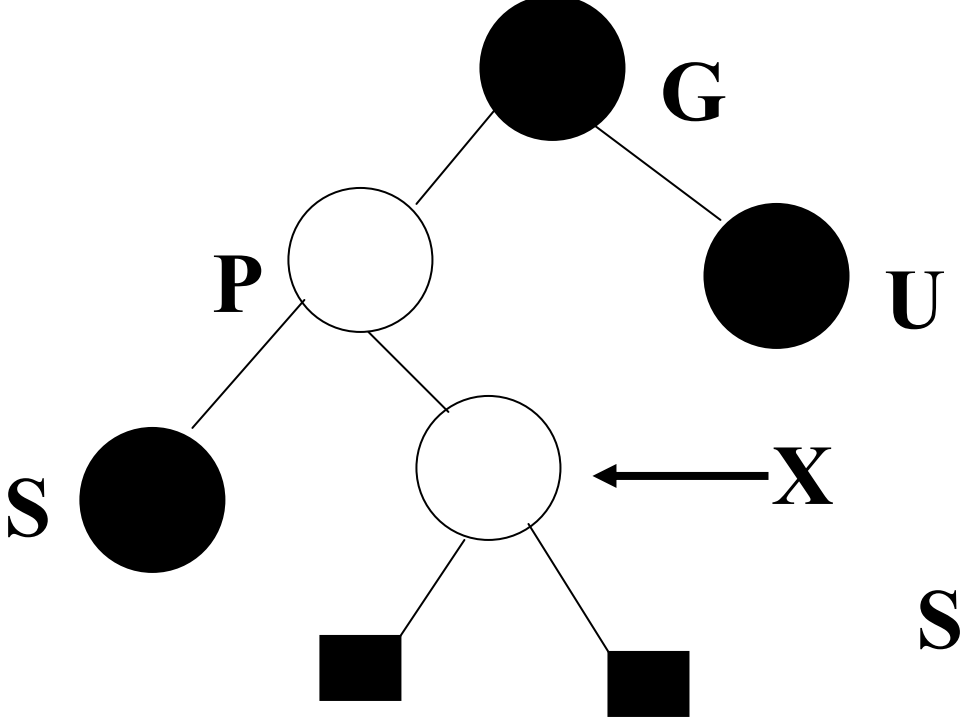
  2 (zig-zag): Nút cha đỏ, chú đen. Nút cha và nút X không cùng là con trái hoặc phải. Thực hiện đổi mầu nút ông thành đỏ, X –đen, quay trái nút cha, quay phải nút ông.

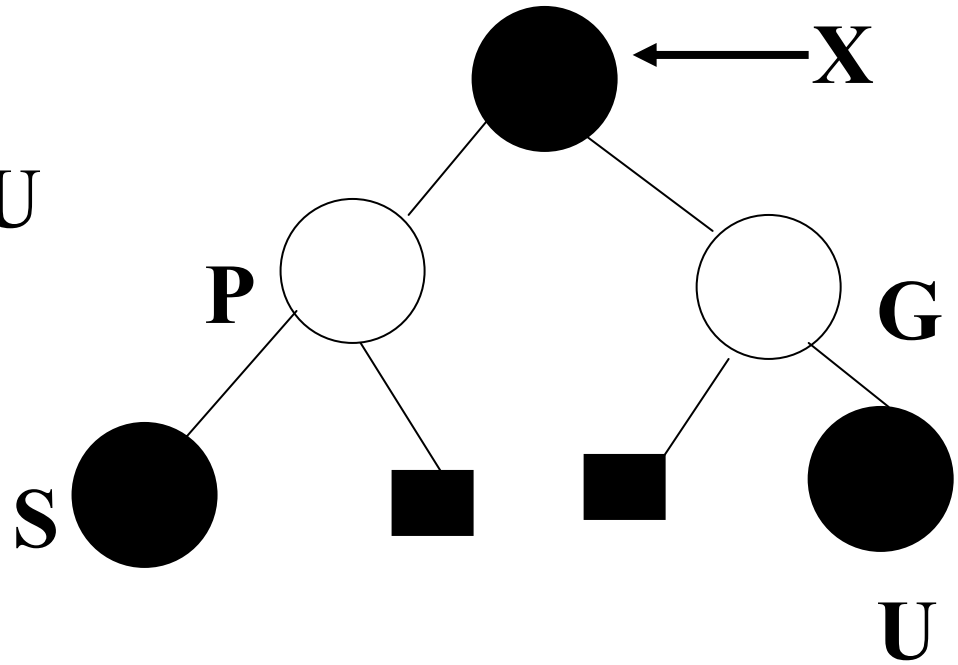
  3 (zig-zig): Nút cha đỏ, chú đen. X và nút cha hoặc cùng là con trái hoặc cùng là con phải. Thực hiện đổi màu nút cha thành đen, ông thành đỏ, quay phải nút ông.

Case 1:



Đổi mầu nút cha, chú thành đen,
mầu nút ông thành đỏ,
X trỏ đến nút ông,
tiếp tục kiểm tra X tại vị trí mới

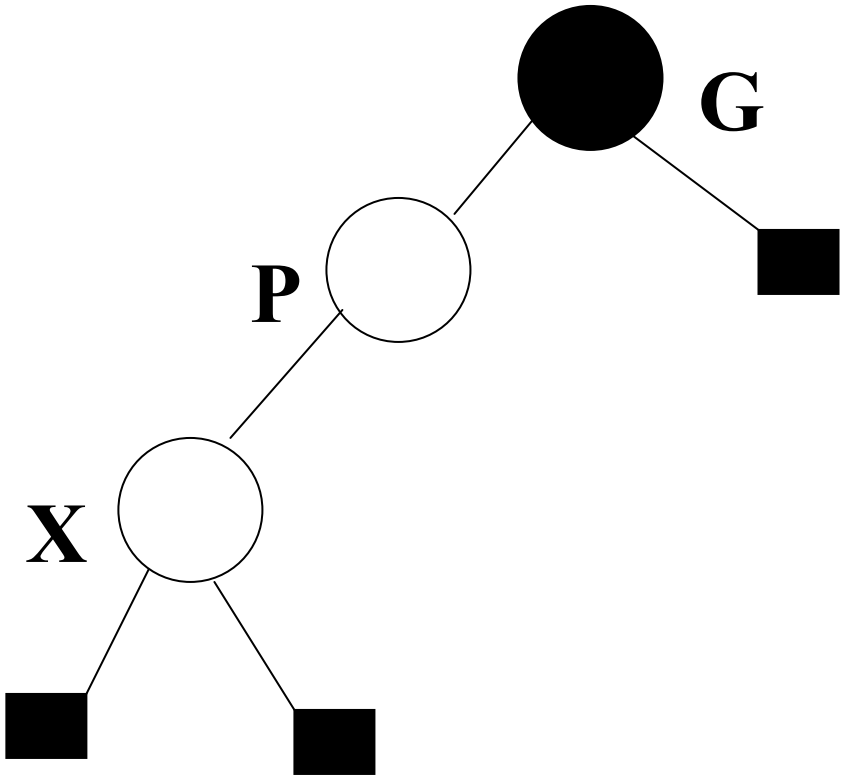


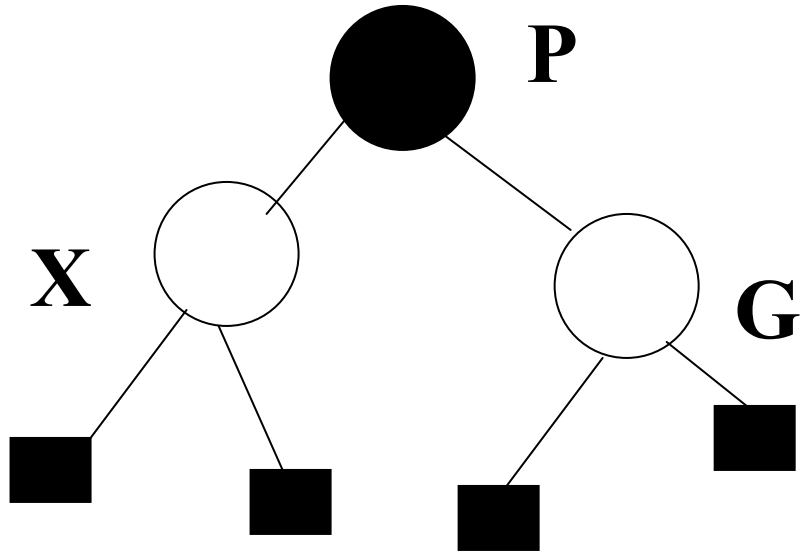


Case 2 – Zig-Zag

Double Rotate
X around P; X around G

Đổi mầu G thành đỏ và X thành đen. (kết thúc)



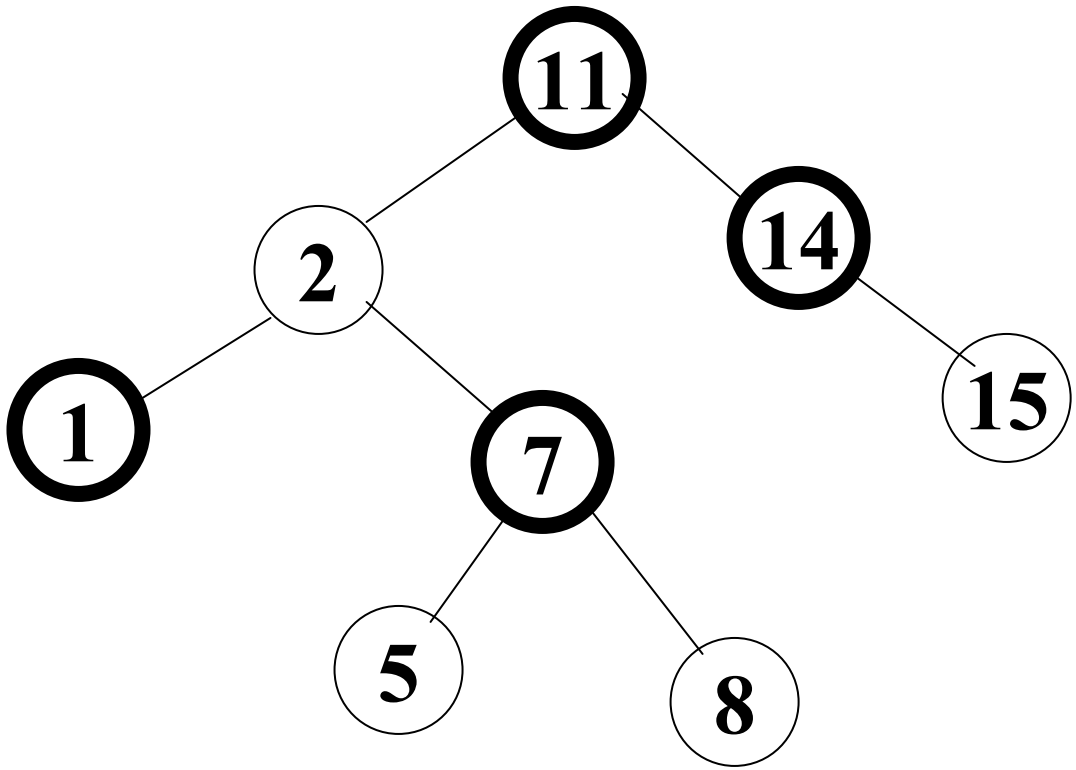


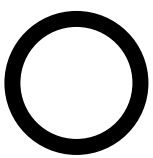
Case 3 – Zig-Zig

Single Rotate P around G

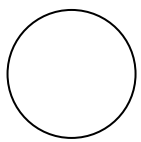
Đặt P - Black and G – Red (kết thúc)

Insert 4 into this R-B Tree





Black node



Red node

Insertion Practice

Insert the values 2, 1, 4, 5, 9, 3, 6, 7 into an initially empty Red-Black Tree

Độ phức tạp của phép Insertion

- O(lg n) – đi từ gốc xuống để tìm vị trí để thêm.
- O(1) - thêm
- O(lg n) – đi ngược lên và hiệu chỉnh màu (case 1)
- Total: O(log n)

Bottom-Up Deletion (BST)

1. Nếu nút cần xóa là nút lá thì thực hiện xóa nút.
2. Nếu nút cần xóa chỉ có một nút con thì kéo nút con lên.
3. Nếu nút cần xóa có cả 2 nút con thì phải tìm nút thay thế.

=> Nút bị xóa thực sự sẽ được thay thế bằng nút có giá trị NULL hoặc nút con bên trái hoặc nút con bên phải.

Bottom-Up Deletion (RBT)

1. Xóa như trên BST.
   - Giả sử U – nút cần xóa
   - Nếu U nút lá, thay thế U bằng con trỏ NULL – kí hiệu là V.
   - Nếu U có một nút con là V, khi xóa thì U được thay bằng V.
   - Nếu U có 2 nút con thì phải tìm nút thay thế - V

Sau khi xóa có thể vi phạm tính chất RB?

1. If U is red?
   => Sau khi xóa U sẽ không vi phạm tính chất đỏ đen.
2. If U is black?
   Nếu U không phải gốc thì khi xóa U sẽ làm thay đổi chiều cao đen trên đường dẫn tương ứng.

Để đảm bảo chiều cao đen ta đẩy một đơn vị đen xuống nút thay thế ta kí hiệu là V.

Vấn đề cần xử lý là V có thể trở thành đen kép (V+) nếu như V đang là nút đen.

Hiệu chỉnh khi xóa nút U – đen và V là đen kép

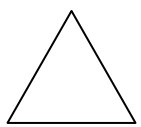
Để V mất tính chất đen kép, ta tìm cách đặt V lên một nút đỏ

- Khi nút V được đặt lên một nút đỏ hoặc nút gốc của cây thì nó trở thành nút đen thường.
- Sau đây chúng ta chỉ xét V là con trái. (Trường hợp V là con phải ta làm tương tự)

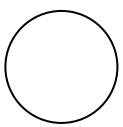
Ký hiệu

- U – nút cần xóa
- V – nút thay thế, V+ đen kép
- P nút cha của V.
- S anh em của V.

Black Node



Red or Black and don’t care

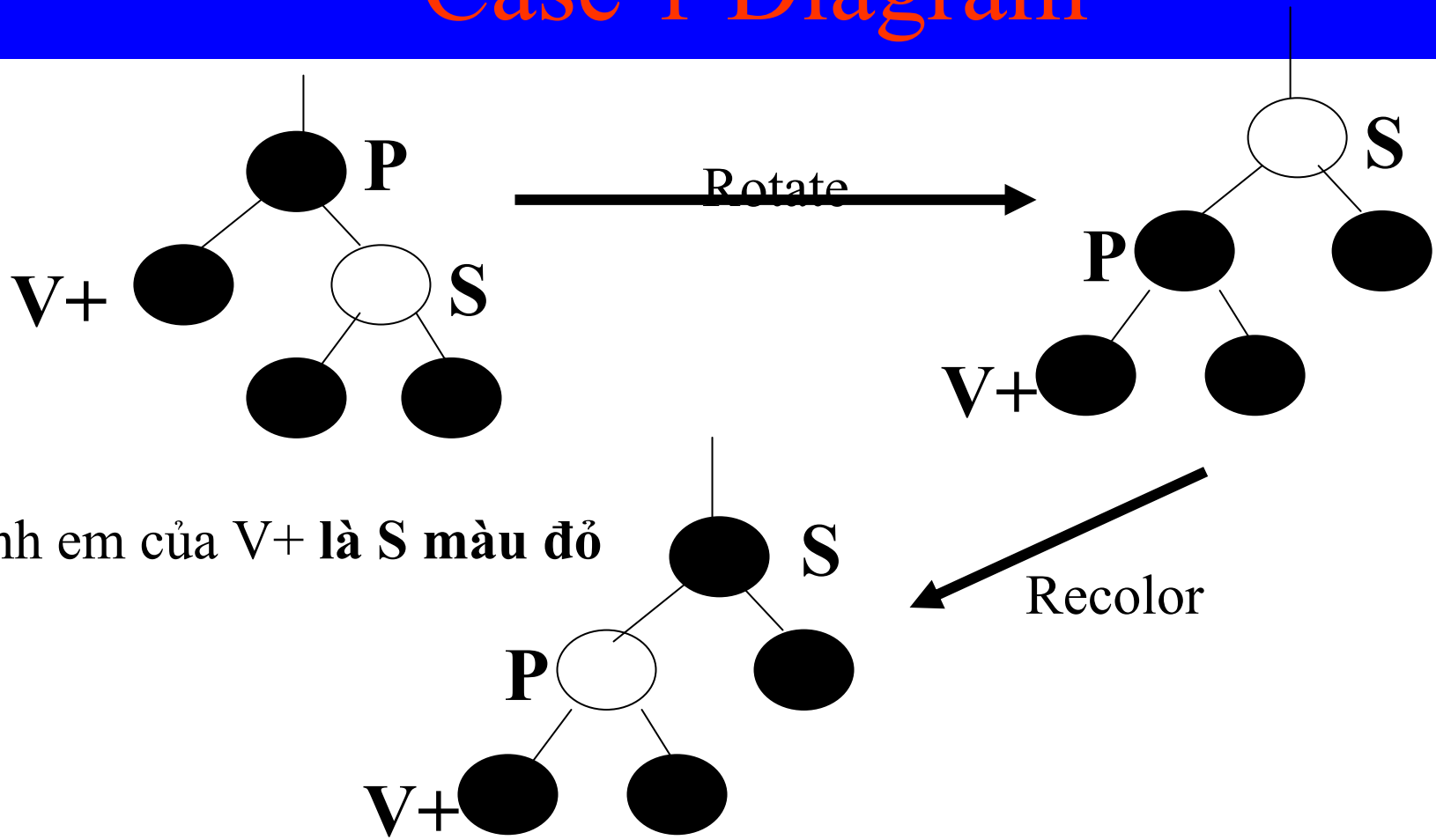


Red Node

Bottom-Up Deletion - Case 1

- Anh em của V+ **là S có màu đỏ**
  - Rotate S around P and recolor S & P
- Chưa kết thúc (NOT a terminal case – One of the other cases will now apply)
- Áp dụng cây nhận được với các trường hợp anh em của V+ là mầu đen.

Case 1 Diagram



•Anh em của V+ **là S màu đỏ**

Rotate S around P and recolor S & P

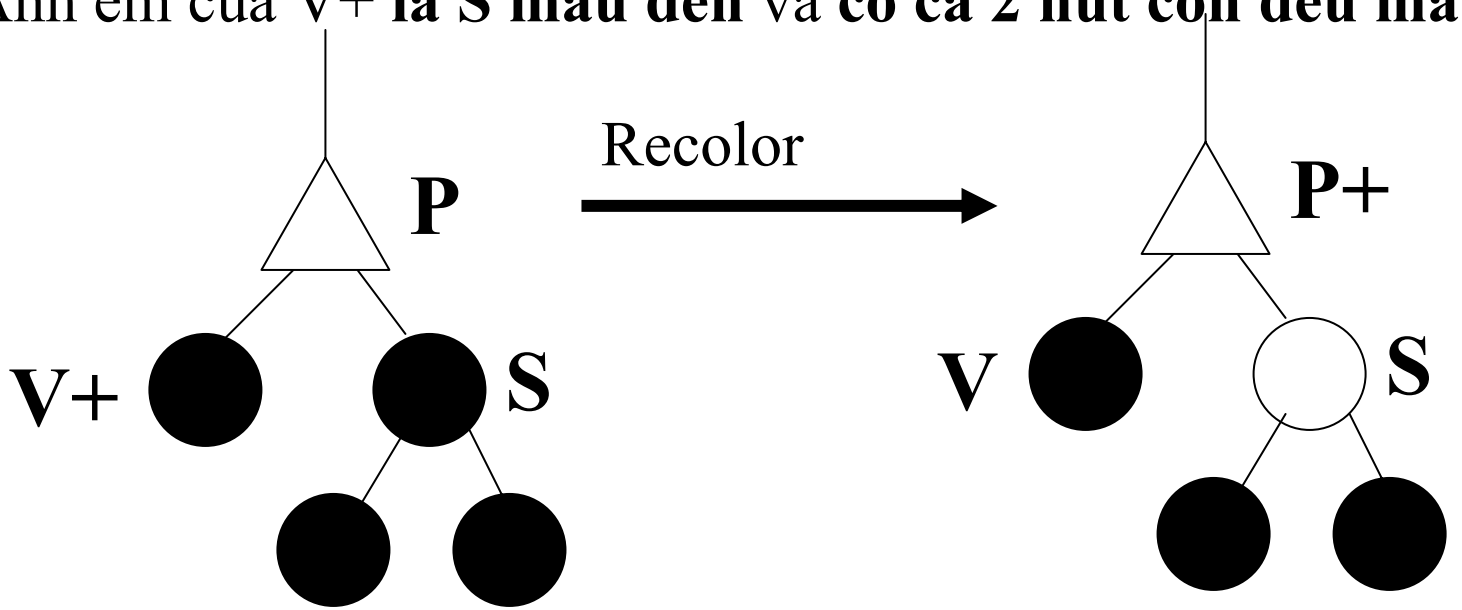
Xét tiếp với V+ ở vị trí mới

Bottom-Up Deletion - Case 2

- Anh em của V+ **là S màu đen** và **có cả 2 nút con đều màu đen**.
  - Đổi mầu S thành đỏ
  - P tiếp nhận một đơn vị đen từ V+:
    - Nếu ban đầu P màu đỏ thì P đổi thành đen và dừng.
    - Nếu P ban đầu màu đen thì P thành đen kép và ta lại phải tiếp tục xét.

Case 2 diagram

•Anh em của V+ **là S màu đen** và **có cả 2 nút con đều màu đen**.



Đổi mầu S thành đỏ.

P tiếp nhận một đơn vị đen từ V+:

- Nếu ban đầu P màu đỏ thì P đổi thành đen và dừng.
- Nếu P ban đầu màu đen thì P thành đen kép và ta lại phải tiếp tục xét.

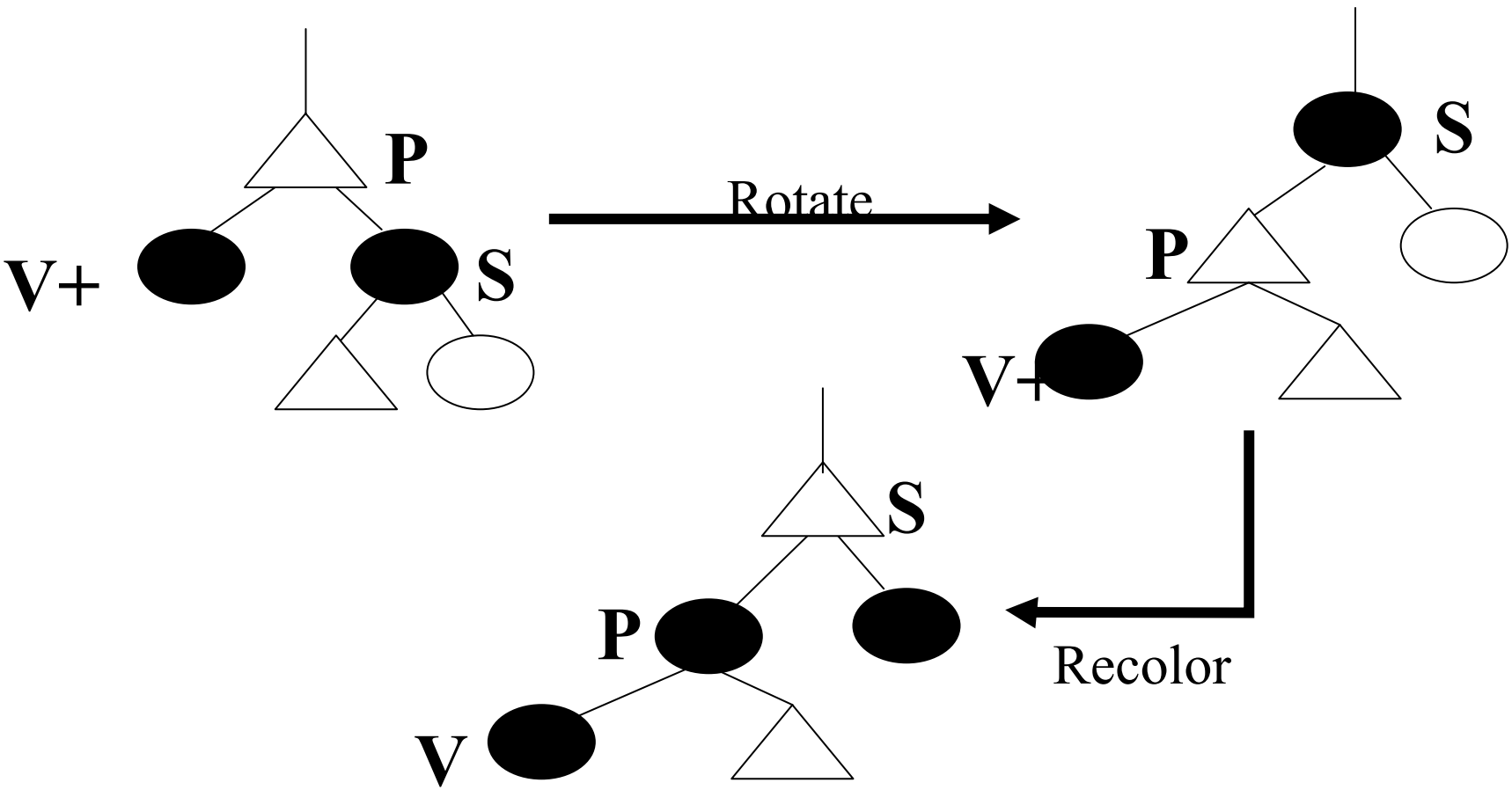
Bottom-Up Deletion - Case 3

- **Nút anh em của V+ là nút S màu đen**
- **Nút S có nút con phải màu đỏ** (nút con trái hoặc màu đỏ hoặc màu đen)
  - Rotate S around P
  - Hoán đổi màu của S và P (Mau(P):=Mau(S); Mau(S):=Mau(P))
  - Đặt màu của con phải của S thành đen
- **Kết thúc**

Case 3 diagrams

Nút anh em của V+ là nút S màu đen

Nút S có nút con phải màu đỏ (con trái hoặc đỏ hoặc đen)



Rotate S around P

Hoán đổi màu của S và P (Mau(P):=Mau(S); Mau(S):=Mau(P))

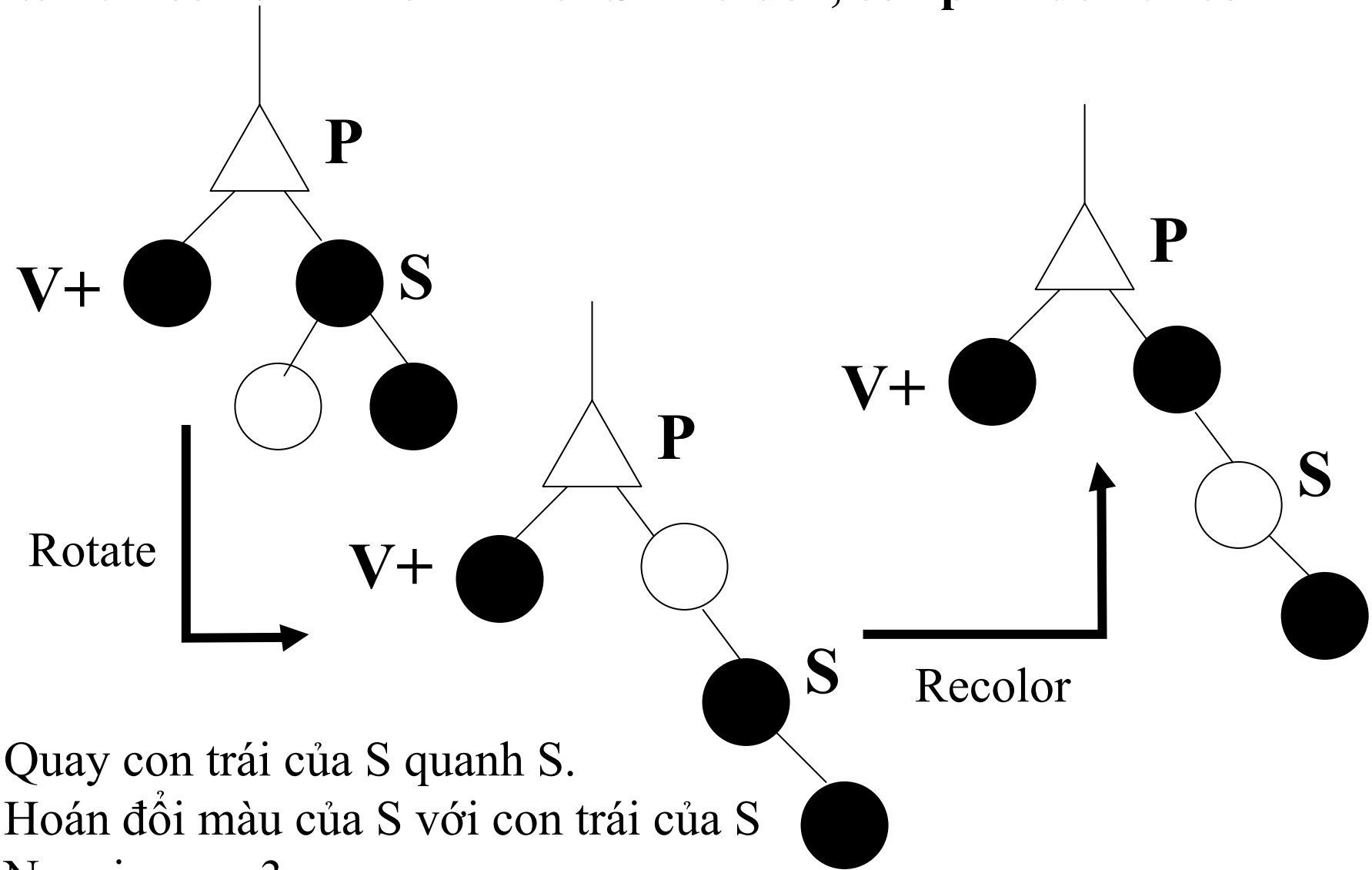
Đặt màu của con phải của S thành đen. (kết thúc)

Bottom-Up Deletion - Case 4

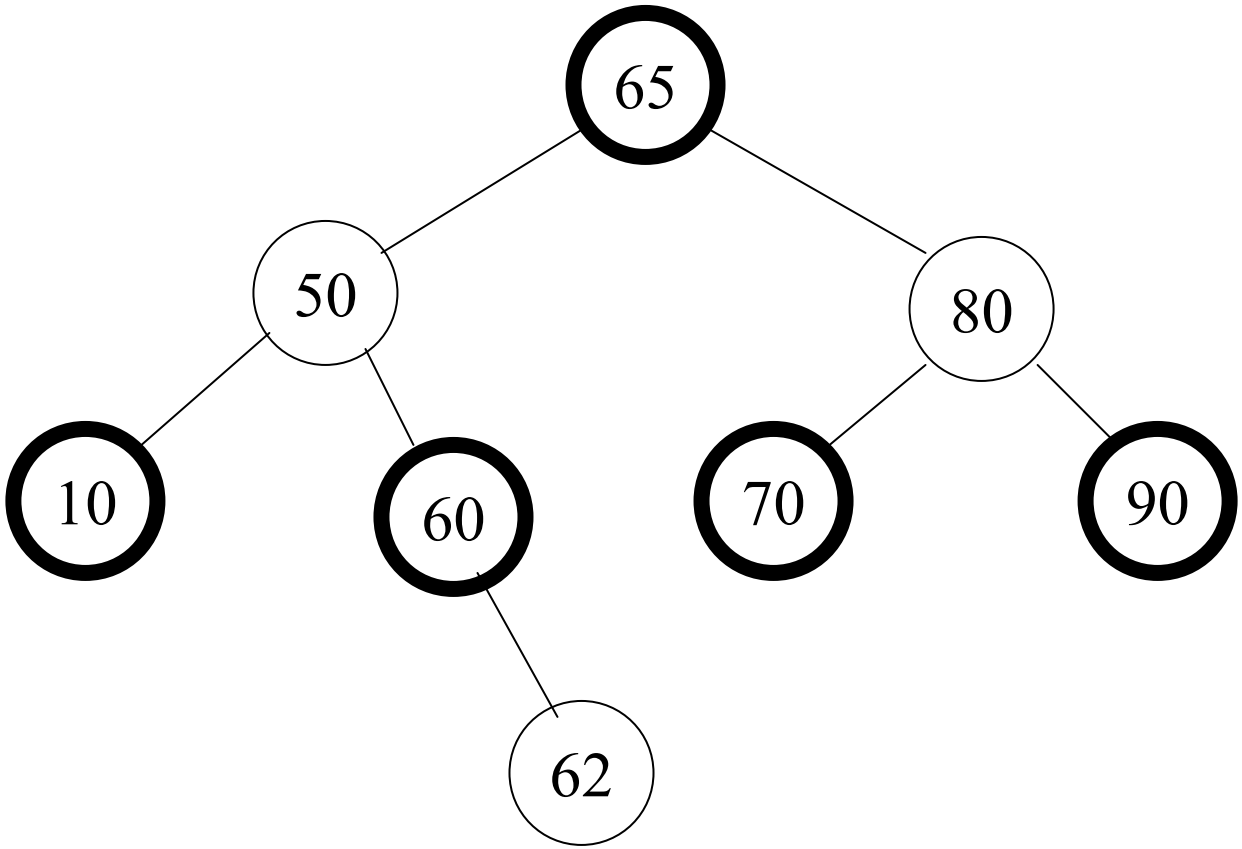
- **Nút V+ có nút anh em là nút S màu đen, nút con phải có màu đen và nút con trái màu đỏ**
  - Quay con trái của S quanh S.
  - Hoán đổi màu của S với màu nút con trái của S
  - Now in case 3

Case 4 Diagrams

- **Nút V+ có nút anh em là nút S màu đen, con phải đen và con trái đỏ**



Quay con trái của S quanh S.
Hoán đổi màu của S với con trái của S
Now in case 3



Perform the following deletions, in the order specified

Delete 90, Delete 80, Delete 70