CSI 2103: Data Structures

HW Assignment 3

Yonsei University
Spring 2022

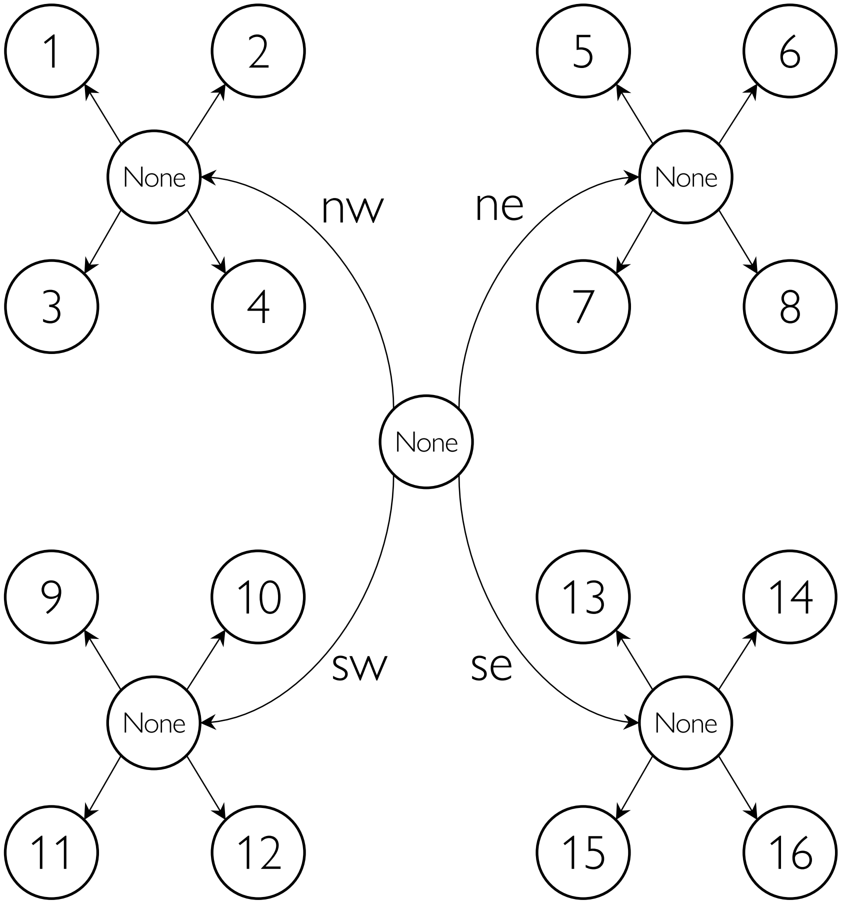
Seong Jae Hwang

Quadtree from Array for HW3

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Quadtree

- At most 4 children
- Our quadtree is always full



column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5	6
y = 1	3	4	7	8
y = 2	9	10	13	14
y = 3	11	12	15	16

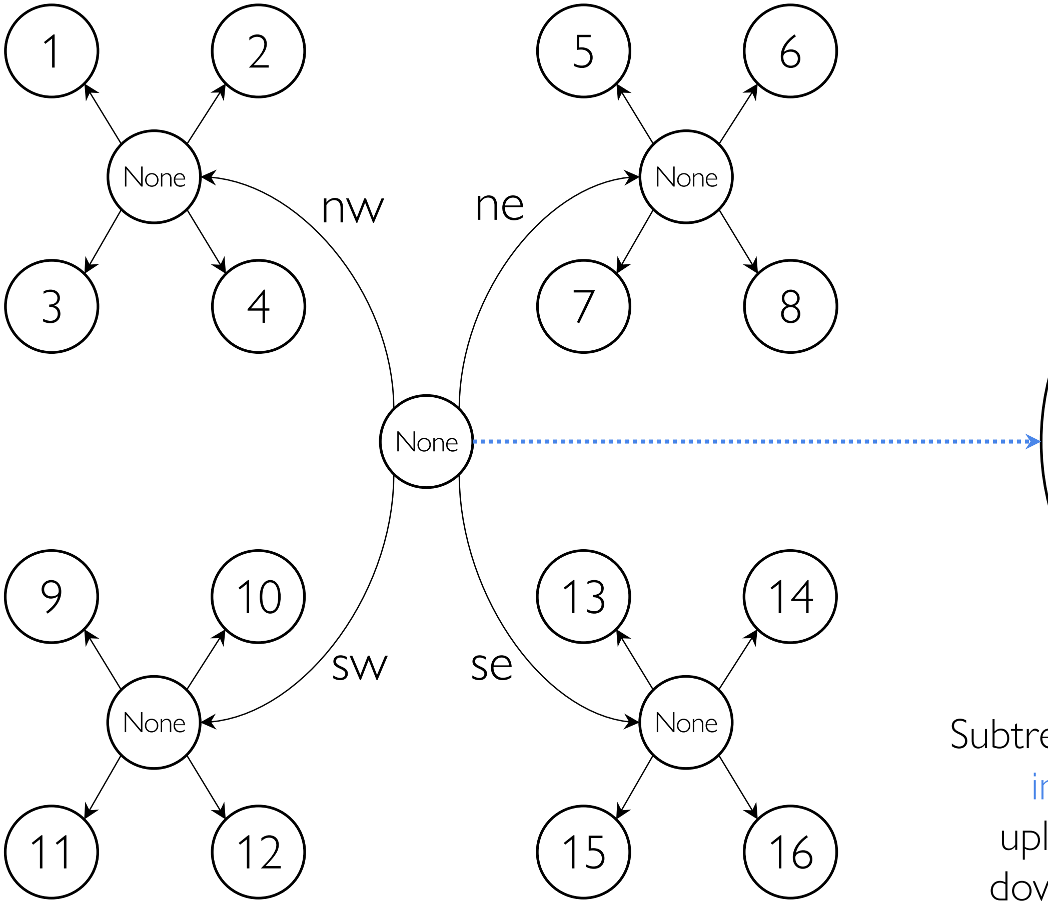
This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Quadtree

- At most 4 children
- Our quadtree is always full

column

row	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5	6
y = 1	3	4	7	8
y = 2	9	10	13	14
y = 3	11	12	15	16



color: None
level: 0

4 children pointers:
{nw, ne, sw, se}

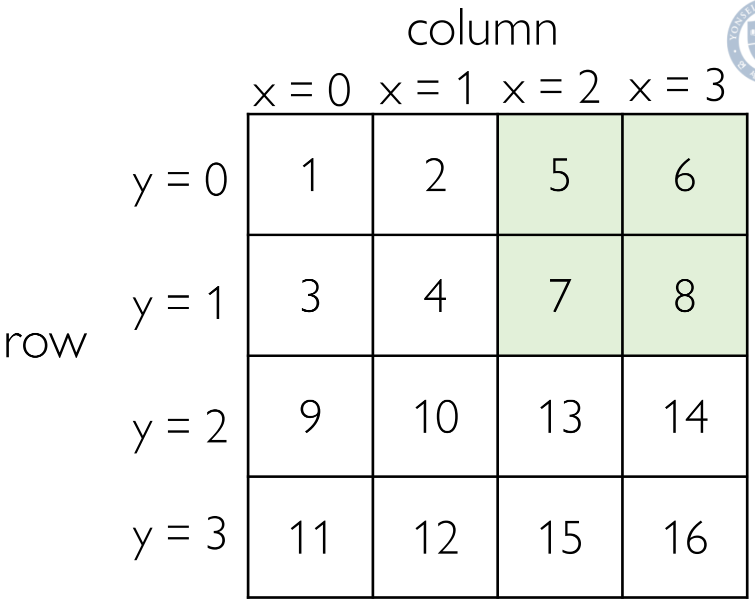
Image area corners:
upleft_x: 0
upleft_y: 0
downright_x: 3
downright_y: 3

Subtree rooted at qt.root covers the entire image area defined by a box with upleft corner at (ul_x=0, ul_y=0) and downright corner at (dr_x=3, dr_y=3)

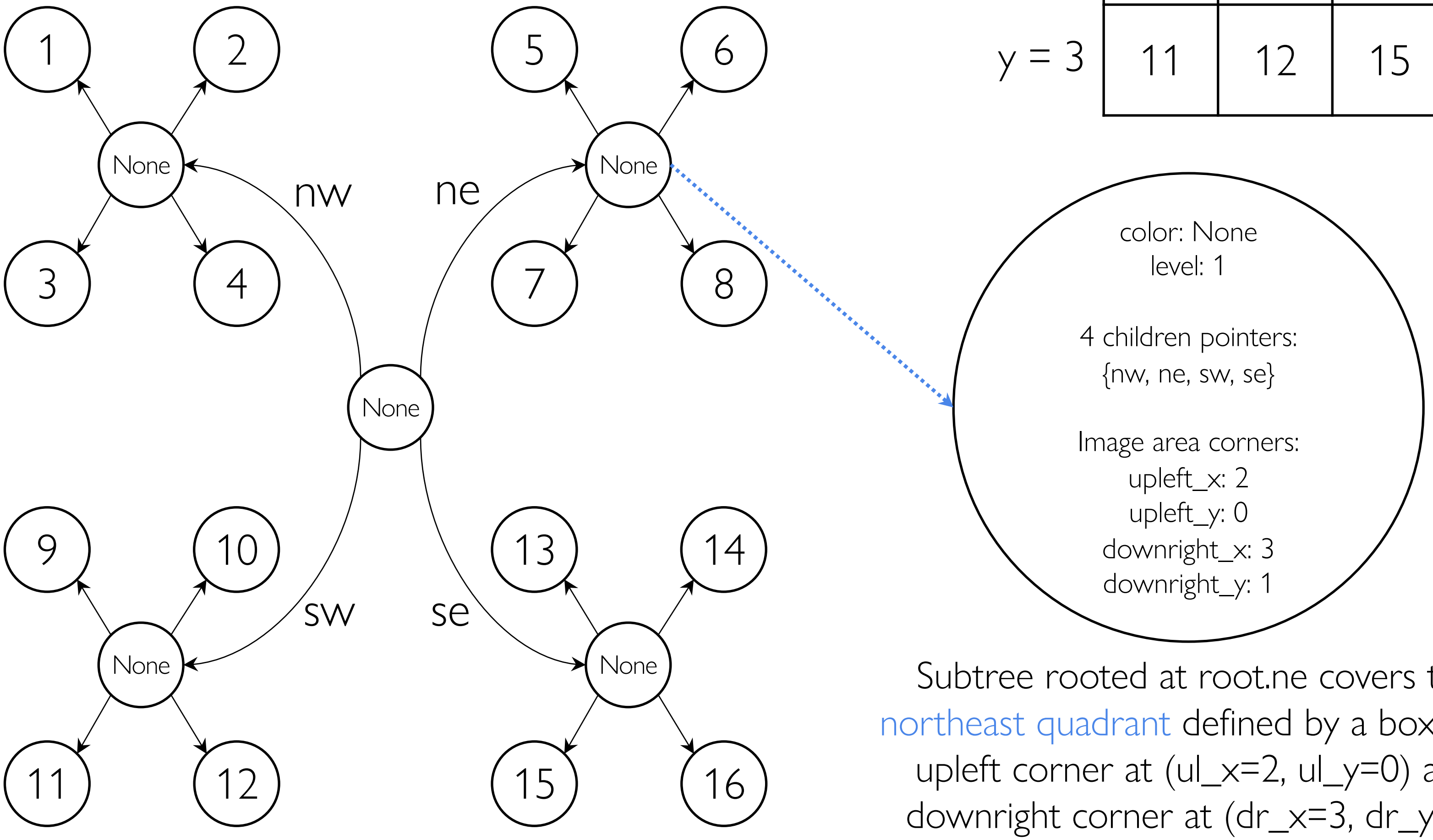
This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Quadtree

- Each child covers its corresponding quadrant with respect to the parent



	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5	6
y = 1	3	4	7	8
y = 2	9	10	13	14
y = 3	11	12	15	16



color: None
level: 1

4 children pointers:
{nw, ne, sw, se}

Image area corners:
upleft_x: 2
upleft_y: 0
downright_x: 3
downright_y: 1

Subtree rooted at root.ne covers the northeast quadrant defined by a box with upleft corner at (ul_x=2, ul_y=0) and downright corner at (dr_x=3, dr_y=1)

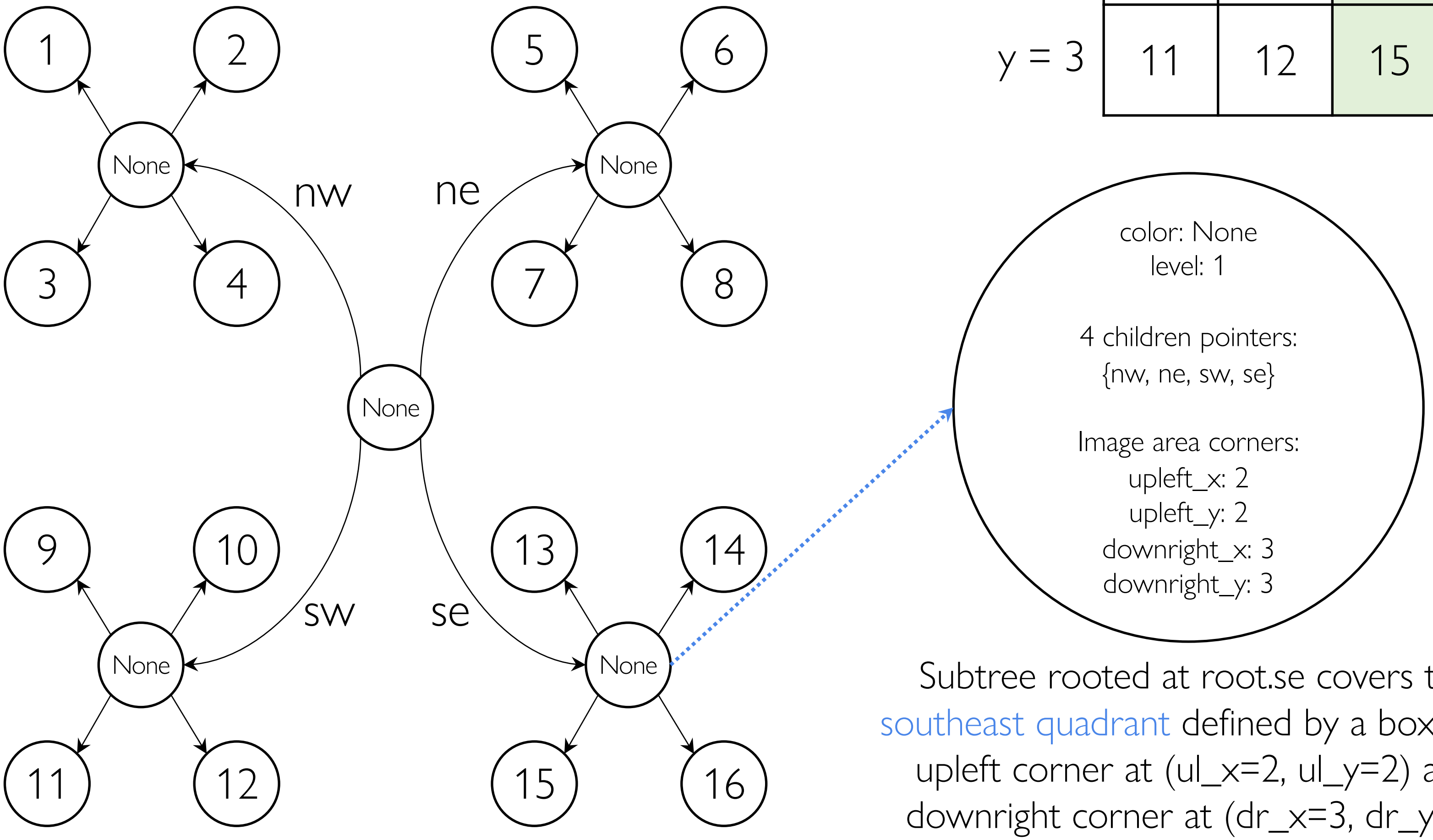
This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Quadtree

- Each child covers its corresponding quadrant with respect to the parent

column

row	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5	6
y = 1	3	4	7	8
y = 2	9	10	13	14
y = 3	11	12	15	16



color: None
level: 1

4 children pointers:
{nw, ne, sw, se}

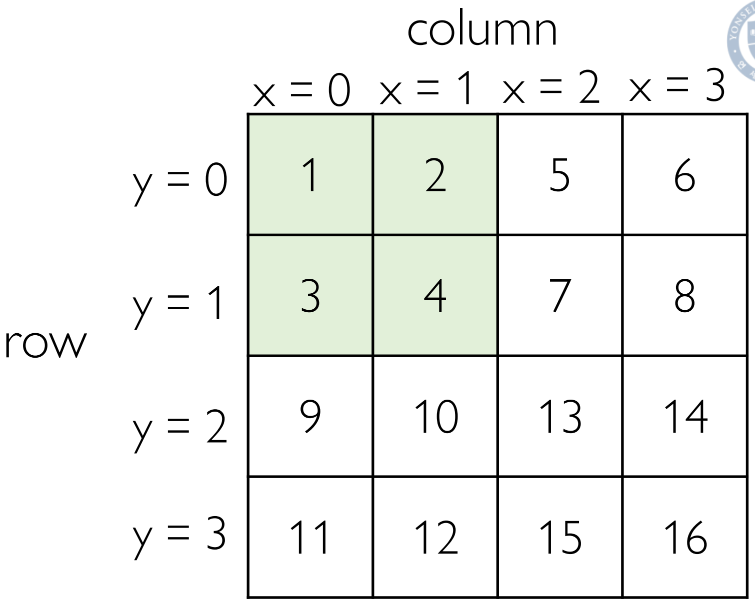
Image area corners:
upleft_x: 2
upleft_y: 2
downright_x: 3
downright_y: 3

Subtree rooted at root.se covers the southeast quadrant defined by a box with upleft corner at (ul_x=2, ul_y=2) and downright corner at (dr_x=3, dr_y=3)

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

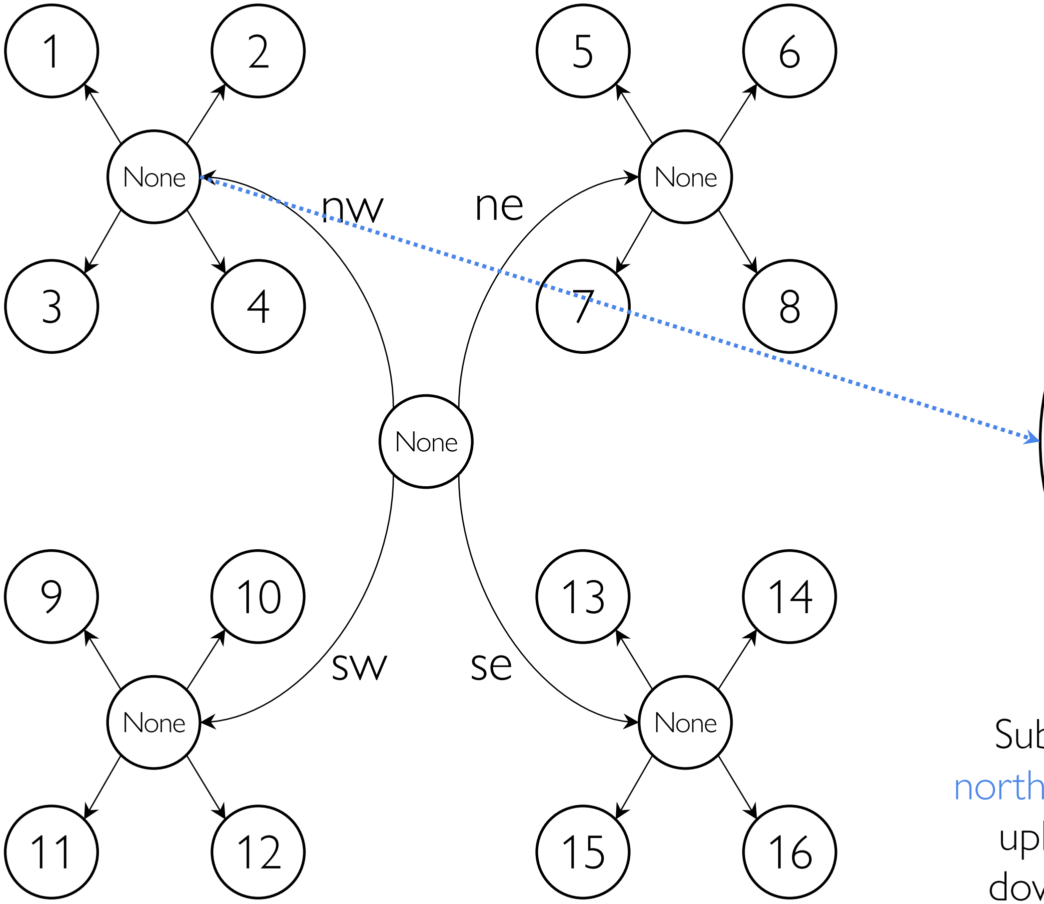
Quadtree

- Each child covers its corresponding quadrant with respect to the parent



column

row	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5	6
y = 1	3	4	7	8
y = 2	9	10	13	14
y = 3	11	12	15	16



color: None
level: 1

4 children pointers:
{nw, ne, sw, se}

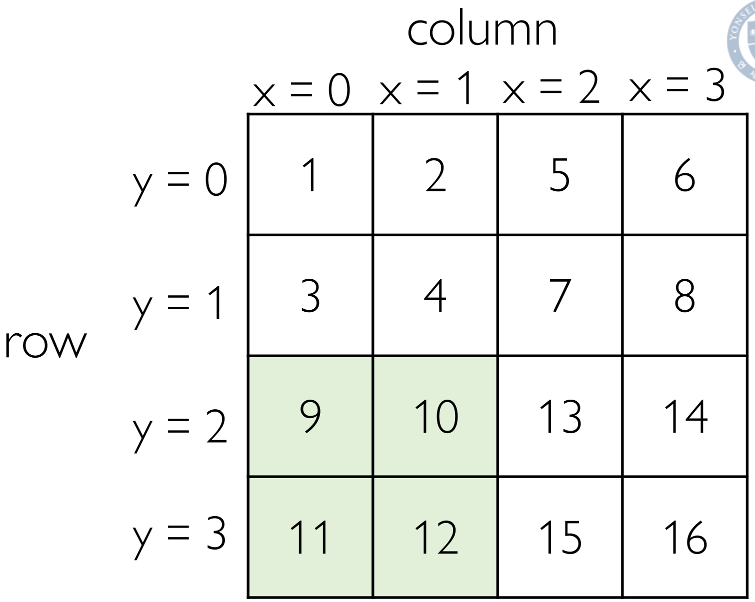
Image area corners:
upleft_x: 0
upleft_y: 0
downright_x: 1
downright_y: 1

Subtree rooted at root.nw covers the northwest quadrant defined by a box with upleft corner at (ul_x=0, ul_y=0) and downright corner at (dr_x=1, dr_y=1)

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Quadtree

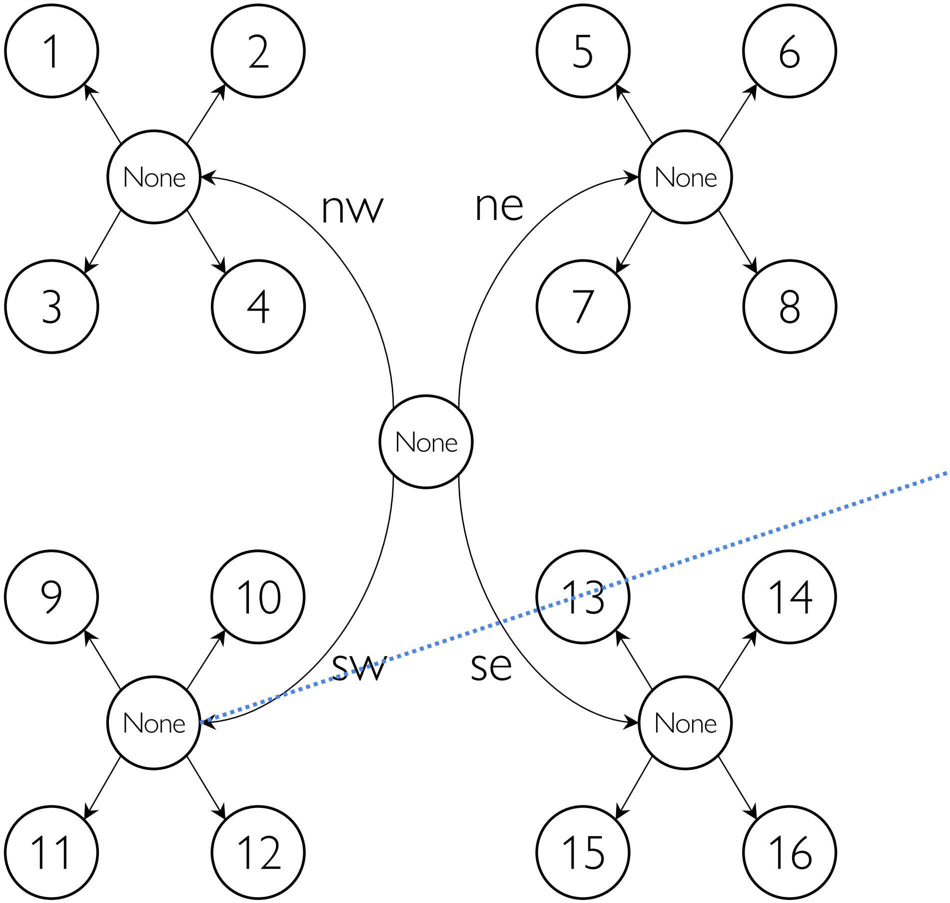
- Each child covers its corresponding quadrant with respect to the parent



column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5	6
y = 1	3	4	7	8
y = 2	9	10	13	14
y = 3	11	12	15	16



color: None
level: 1

4 children pointers:
{nw, ne, sw, se}

Image area corners:
upleft_x: 0
upleft_y: 2
downright_x: 1
downright_y: 3

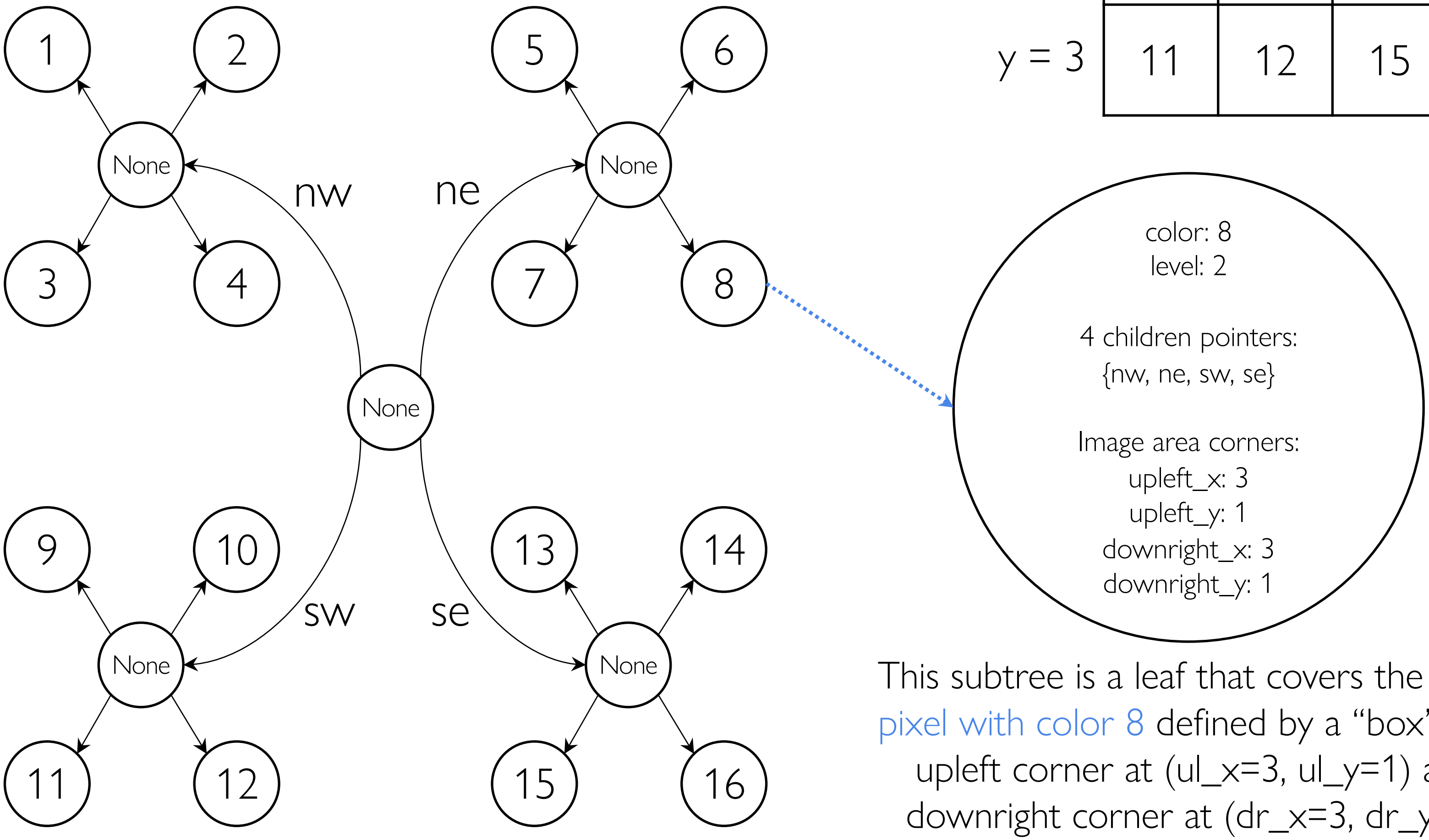
Subtree rooted at root.sw covers the southwest quadrant defined by a box with upleft corner at (ul_x=0, ul_y=2) and downright corner at (dr_x=1, dr_y=3)

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Quadtree

- A leaf is a node which covers only a pixel. A leaf always has a color.

		column			
		x = 0	x = 1	x = 2	x = 3
row	y = 0	1	2	5	6
	y = 1	3	4	7	8
	y = 2	9	10	13	14
	y = 3	11	12	15	16



color: 8
level: 2

4 children pointers:
{nw, ne, sw, se}

Image area corners:
upleft_x: 3
upleft_y: 1
downright_x: 3
downright_y: 1

This subtree is a leaf that covers the single pixel with color 8 defined by a “box” with upleft corner at (ul_x=3, ul_y=1) and downright corner at (dr_x=3, dr_y=1)

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

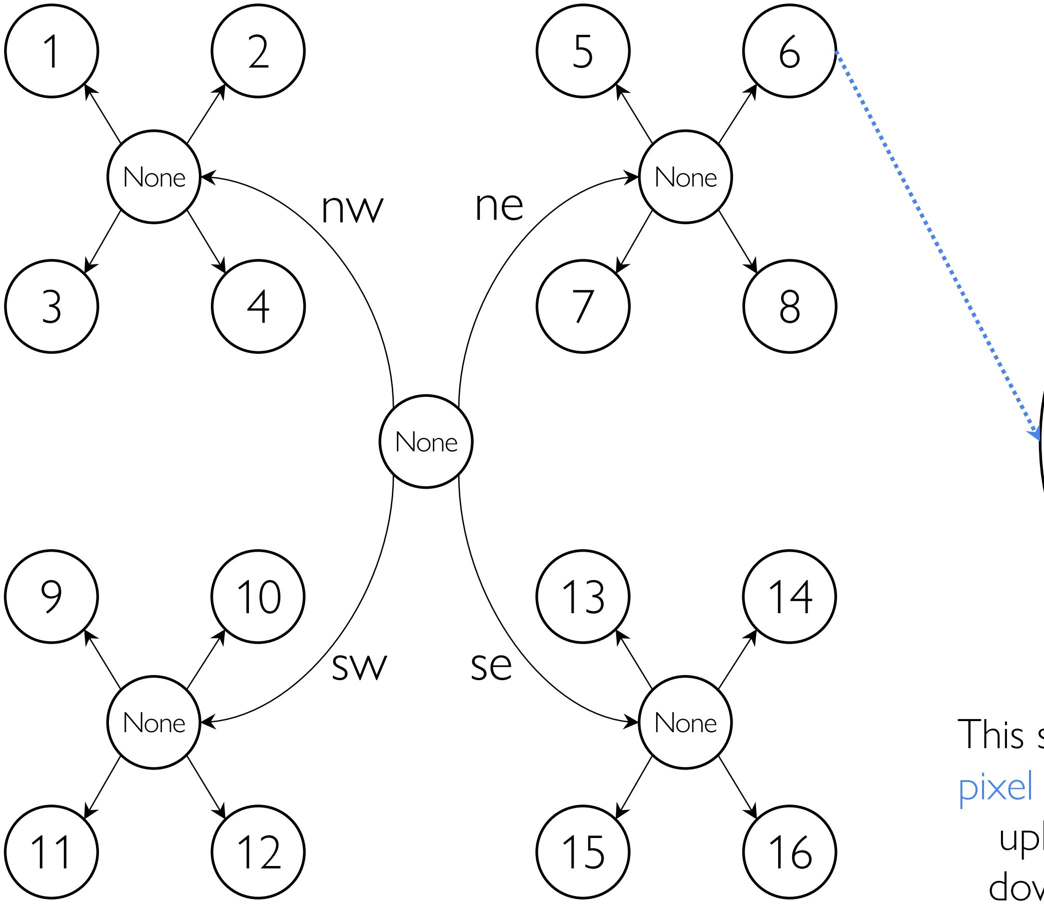
Quadtree

- A leaf is a node which covers only a pixel. A leaf always has a color.

column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5	6
y = 1	3	4	7	8
y = 2	9	10	13	14
y = 3	11	12	15	16



color: 6
level: 2

4 children pointers:
{nw, ne, sw, se}

Image area corners:
upleft_x: 3
upleft_y: 0
downright_x: 3
downright_y: 0

This subtree is a leaf that covers the single pixel with color 8 defined by a "box" with upleft corner at (ul_x=3, ul_y=0) and downright corner at (dr_x=3, dr_y=0)

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Quadtree Class

- Contains the image as self.image and its corresponding quadtree rooted at self.root

- self.image
    - the image array needs to be accessed throughout the recursive calls
    - our functions pass the Quadtree variable
- self.root
    - the actual quadtree begins from the root, so we just keep the root node

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Part 1A: Build a Quadtree from Array

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

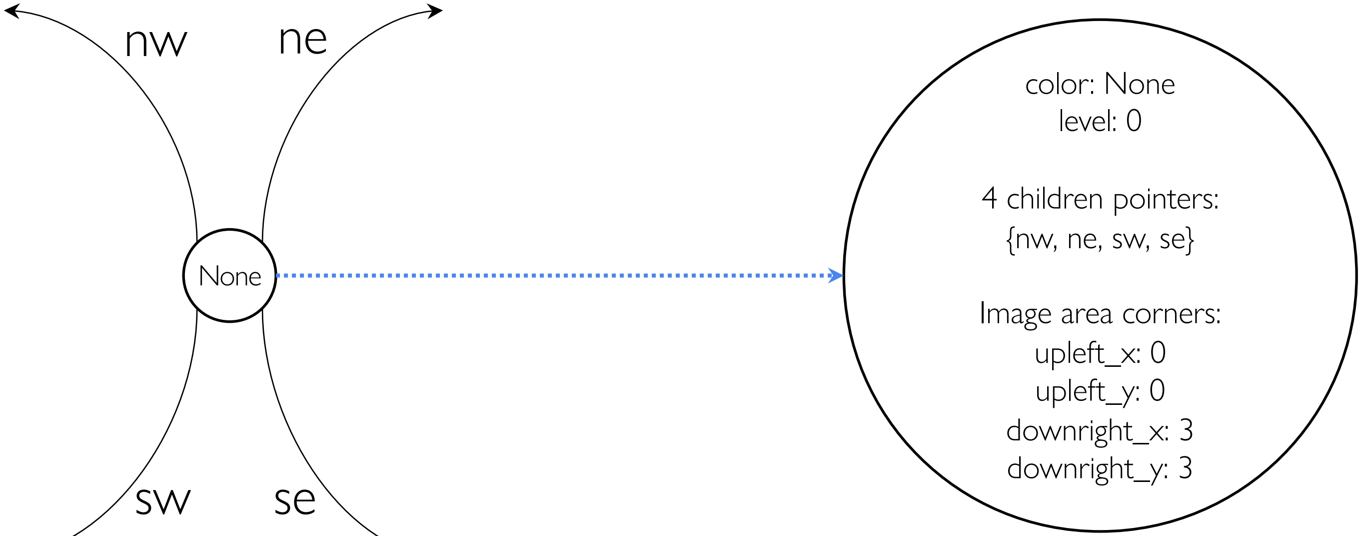
Build a Quadtree

- Start from the root node with proper corners (it's a full image)

column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5	6
y = 1	3	4	7	8
y = 2	9	10	13	14
y = 3	11	12	15	16



color: None
level: 0

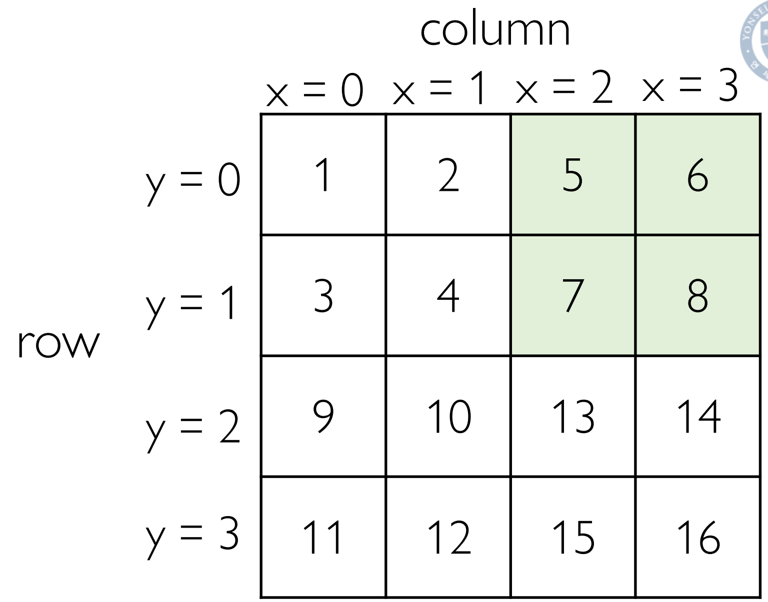
4 children pointers:
{nw, ne, sw, se}

Image area corners:
upleft_x: 0
upleft_y: 0
downright_x: 3
downright_y: 3

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Build a Quadtree

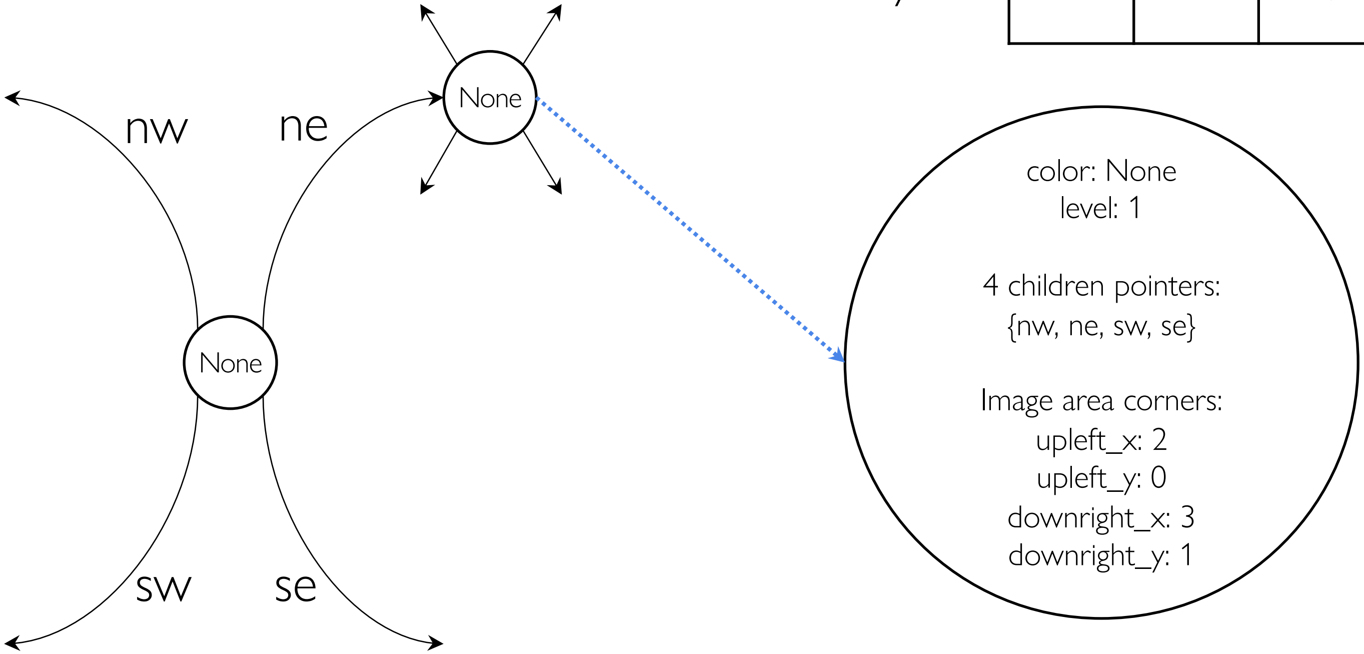
- Recursively create children nodes with proper corners and levels
  - corners can be computed using compute_corners function



column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5	6
y = 1	3	4	7	8
y = 2	9	10	13	14
y = 3	11	12	15	16



color: None
level: 1

4 children pointers:
{nw, ne, sw, se}

Image area corners:
upleft_x: 2
upleft_y: 0
downright_x: 3
downright_y: 1

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

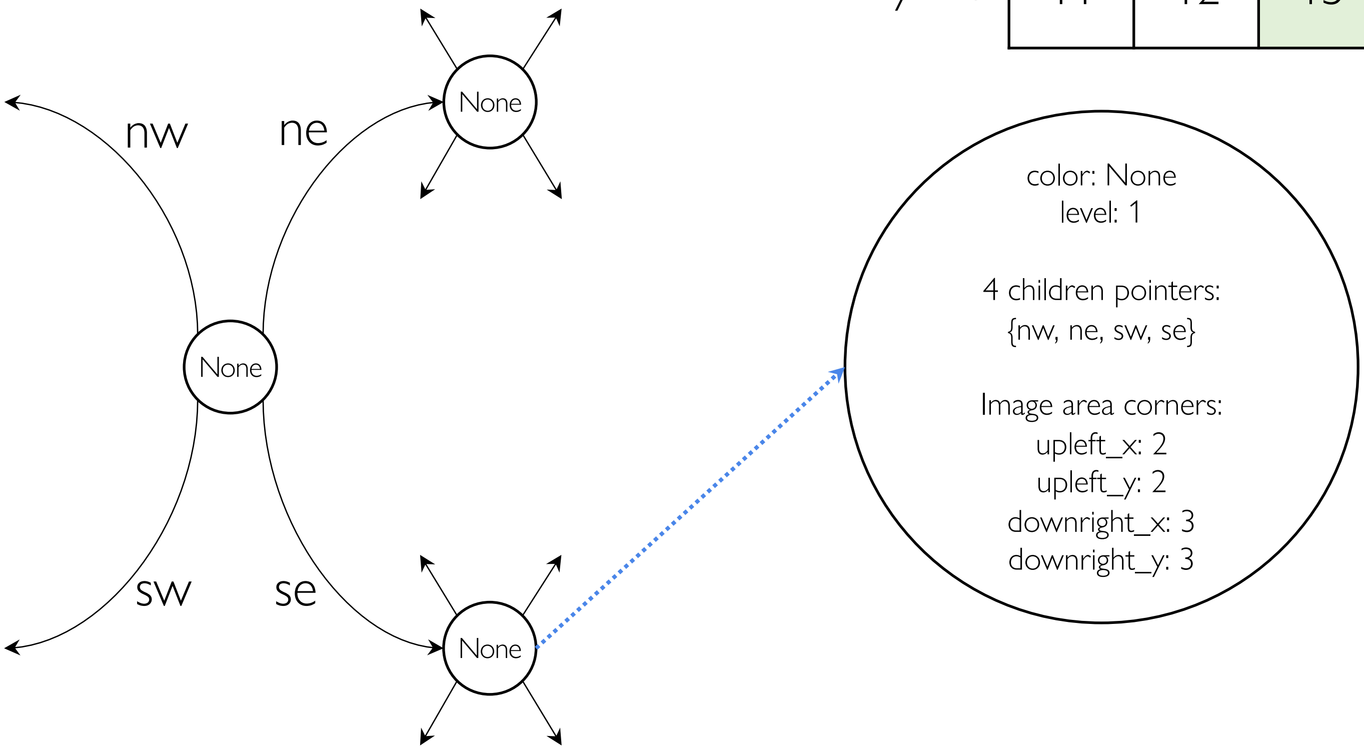
Build a Quadtree

- Recursively create children nodes with proper corners and levels
  - corners can be computed using compute_corners function

column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5	6
y = 1	3	4	7	8
y = 2	9	10	13	14
y = 3	11	12	15	16



color: None
level: 1

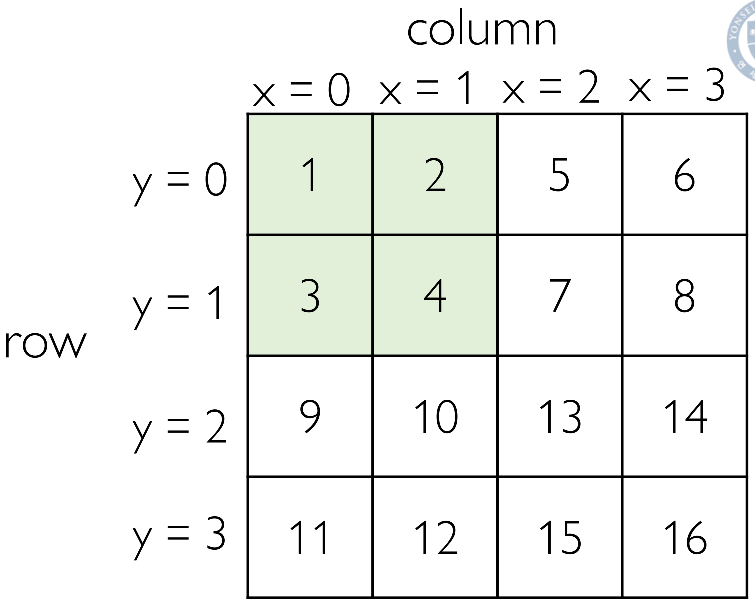
4 children pointers:
{nw, ne, sw, se}

Image area corners:
upleft_x: 2
upleft_y: 2
downright_x: 3
downright_y: 3

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Build a Quadtree

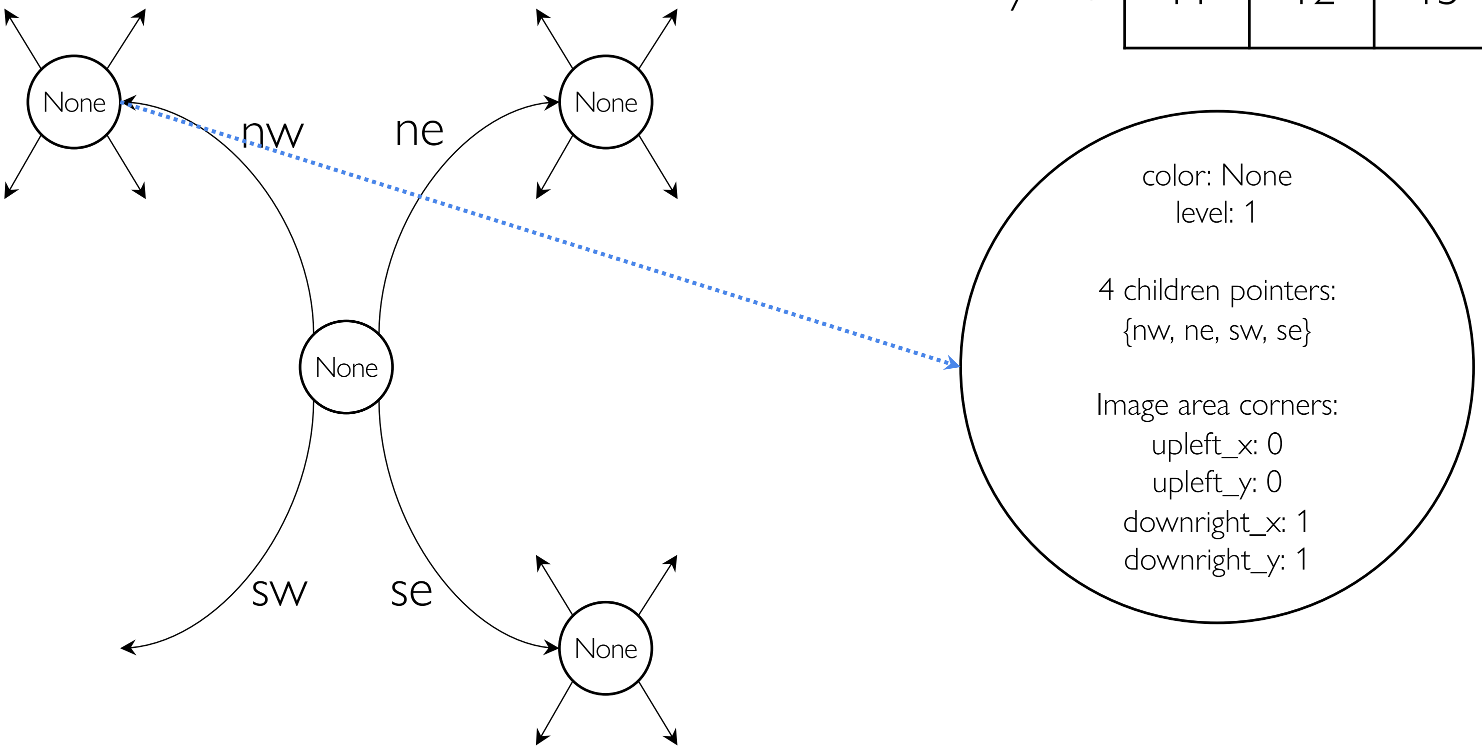
- Recursively create children nodes with proper corners and levels
  - corners can be computed using compute_corners function



column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5	6
y = 1	3	4	7	8
y = 2	9	10	13	14
y = 3	11	12	15	16



color: None
level: 1

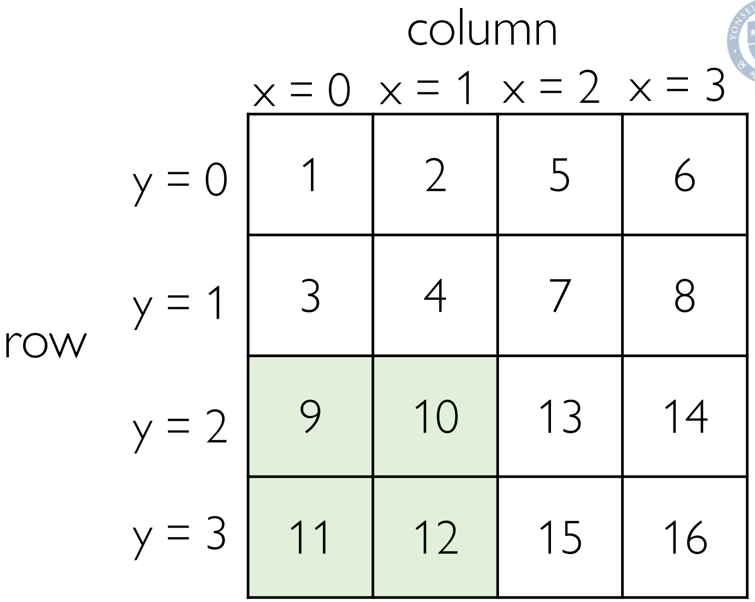
4 children pointers:
{nw, ne, sw, se}

Image area corners:
upleft_x: 0
upleft_y: 0
downright_x: 1
downright_y: 1

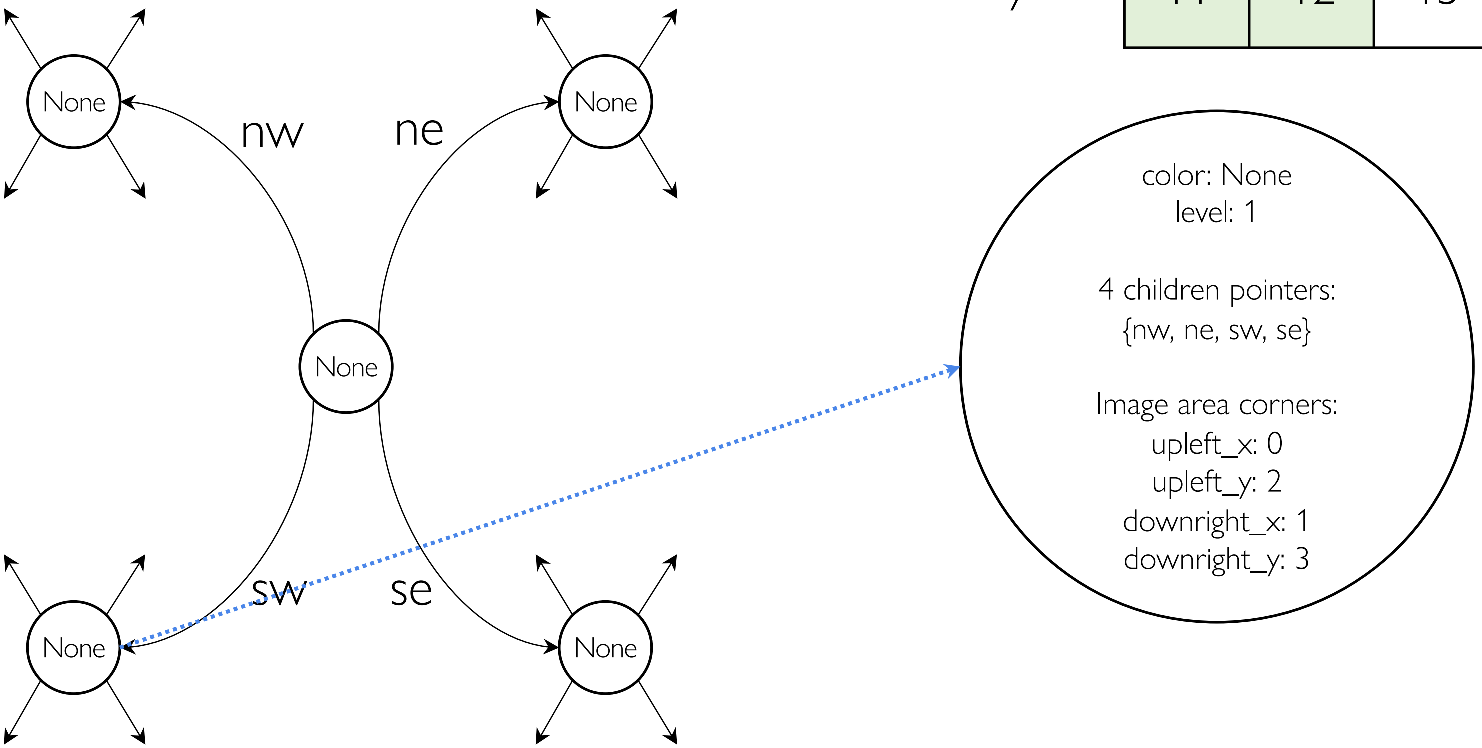
This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Build a Quadtree

- Recursively create children nodes with proper corners and levels
    - corners can be computed using compute_corners function



		column			
		x = 0	x = 1	x = 2	x = 3
row	y = 0	1	2	5	6
	y = 1	3	4	7	8
	y = 2	9	10	13	14
	y = 3	11	12	15	16



color: None
level: 1

4 children pointers:
{nw, ne, sw, se}

Image area corners:
upleft_x: 0
upleft_y: 2
downright_x: 1
downright_y: 3

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

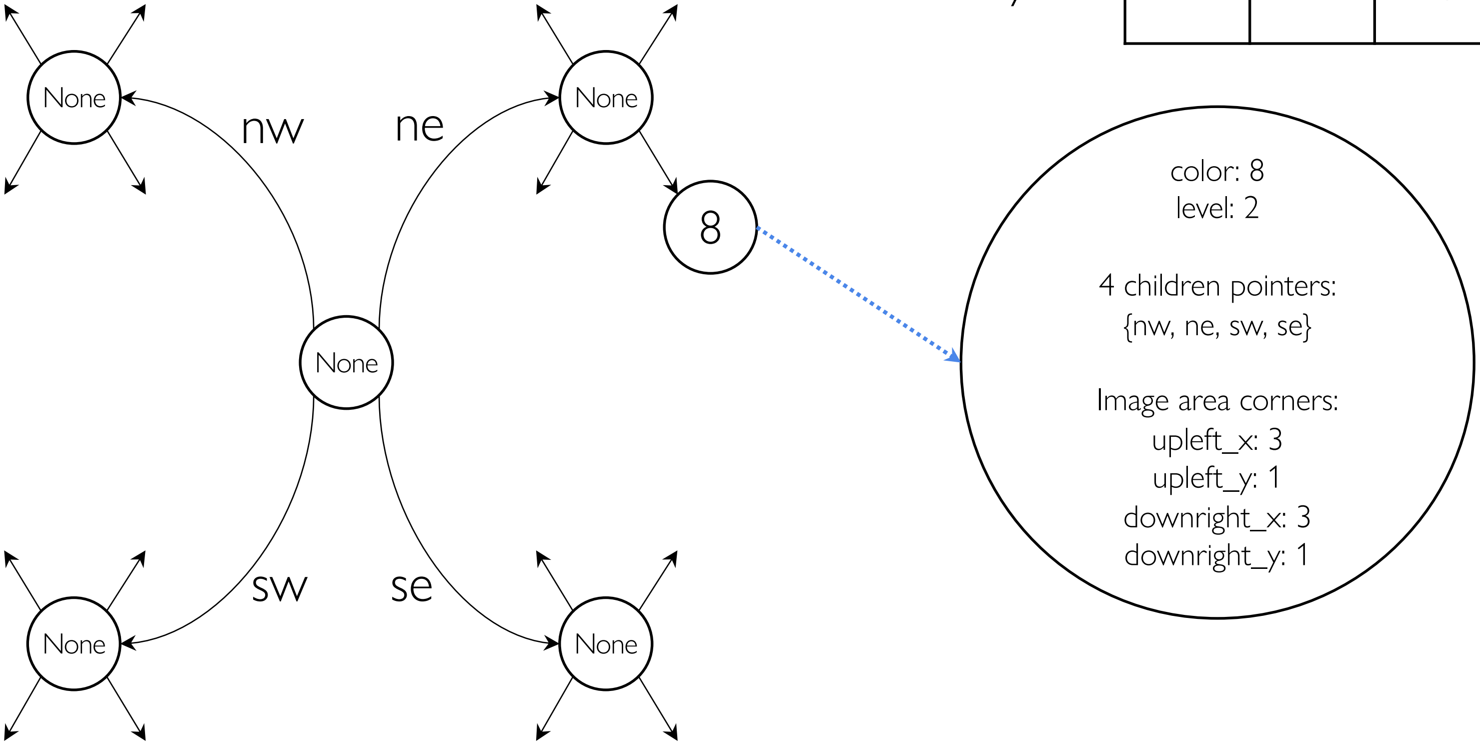
Build a Quadtree

- If you find corners that imply it's a single pixel, assign colors and treat them as leaves

column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5	6
y = 1	3	4	7	8
y = 2	9	10	13	14
y = 3	11	12	15	16



color: 8
level: 2

4 children pointers:
{nw, ne, sw, se}

Image area corners:
upleft_x: 3
upleft_y: 1
downright_x: 3
downright_y: 1

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

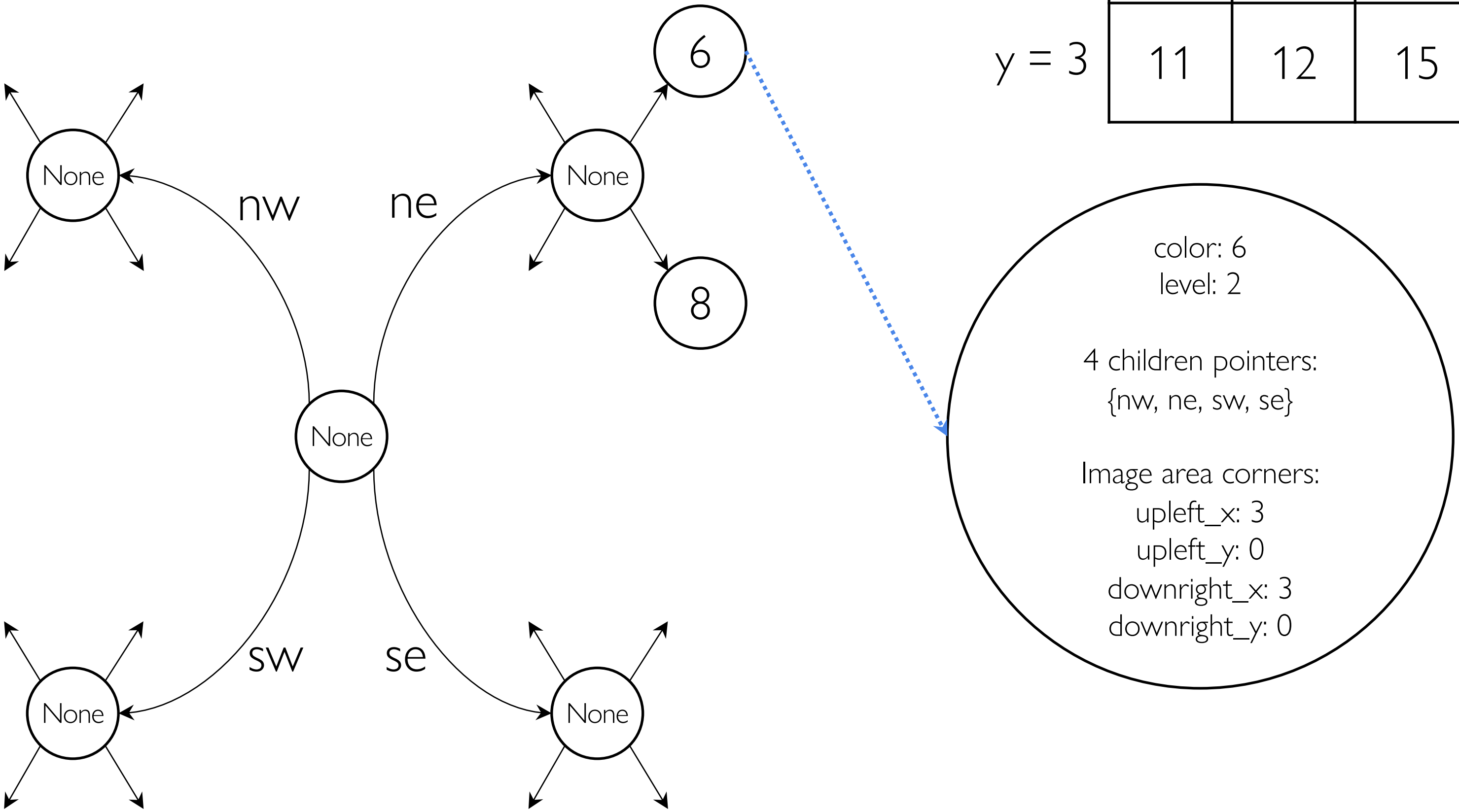
Build a Quadtree

- If you find corners that imply it's a single pixel, assign colors and treat them as leaves

column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5	6
y = 1	3	4	7	8
y = 2	9	10	13	14
y = 3	11	12	15	16



color: 6
level: 2

4 children pointers:
{nw, ne, sw, se}

Image area corners:
upleft_x: 3
upleft_y: 0
downright_x: 3
downright_y: 0

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

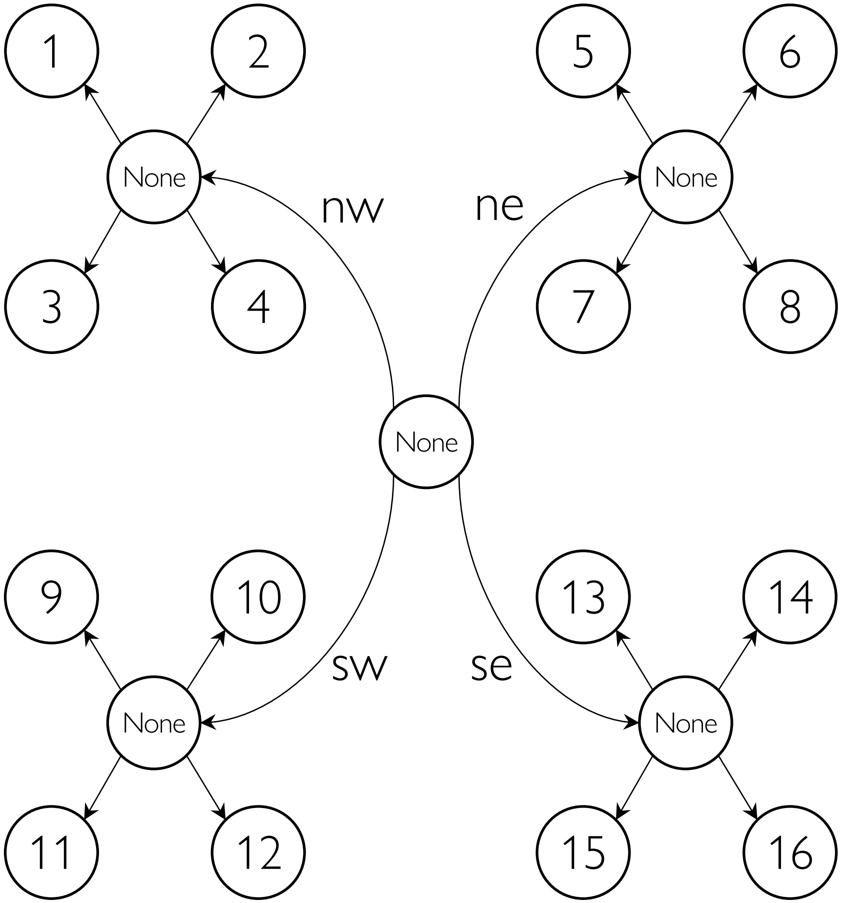
Build a Quadtree

- Eventually all single pixels will be reached and the quadtree will be constructed

column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5	6
y = 1	3	4	7	8
y = 2	9	10	13	14
y = 3	11	12	15	16



This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Build a Quadtree

- build_quadtree calls build_quadtree_recursive
- The code itself is very short
- color is important, but the corners are also important
  - read and use compute_image_corners
    - ex: Node(None, compute_image_corners(node, ‘nw’), node.level+1)
- In Part 1A, you also implement two simple helper functions:
  - count_nodes: recursively count the # of nodes in a quadtree
  - max_num_nodes: mathematically compute the maximum # of nodes you can have in a quadtree built from an N by N image

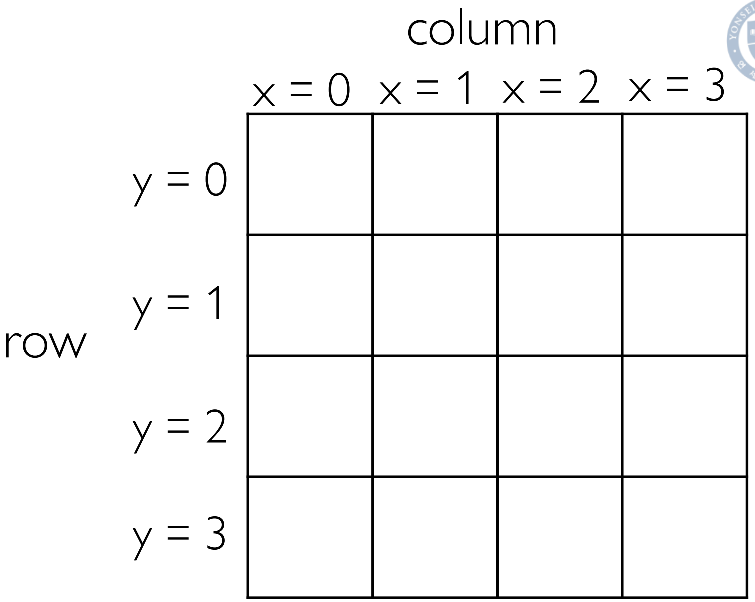
This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

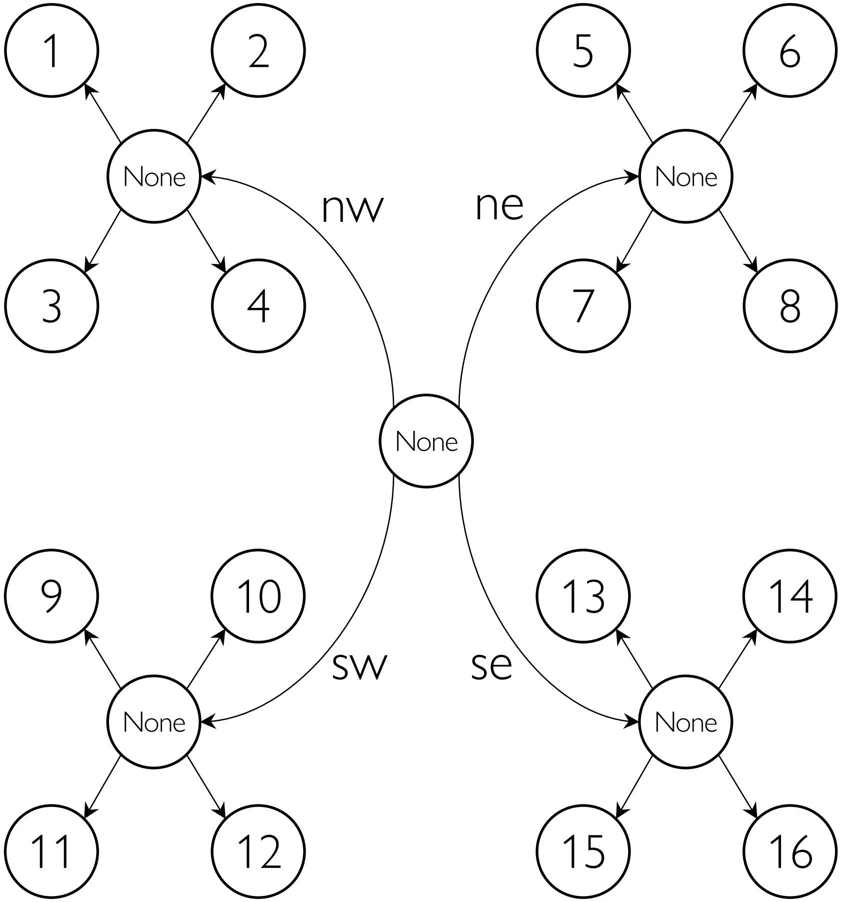
Part 1B: Reconstruct

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Quadtree to Image

- Traverse the quadtree and assign color to corresponding area with upleft corner at (ul_x, ul_y) and downright corner at (dr_x, dr_y)

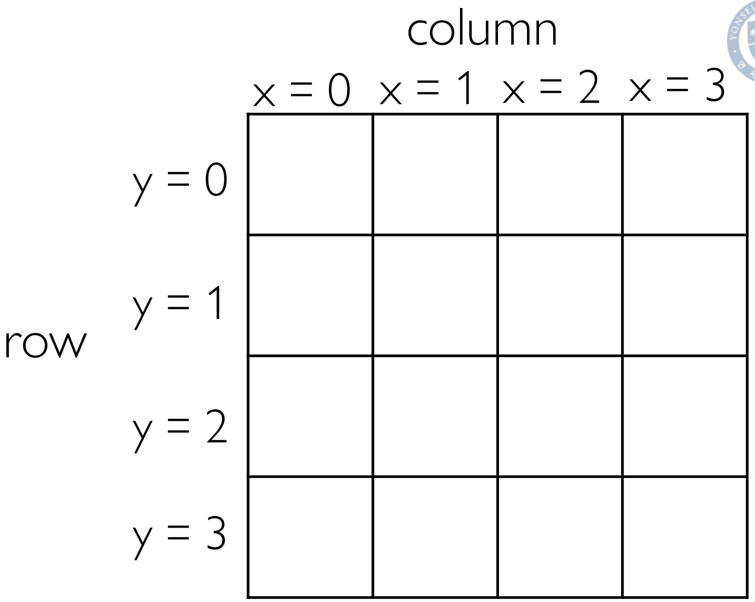


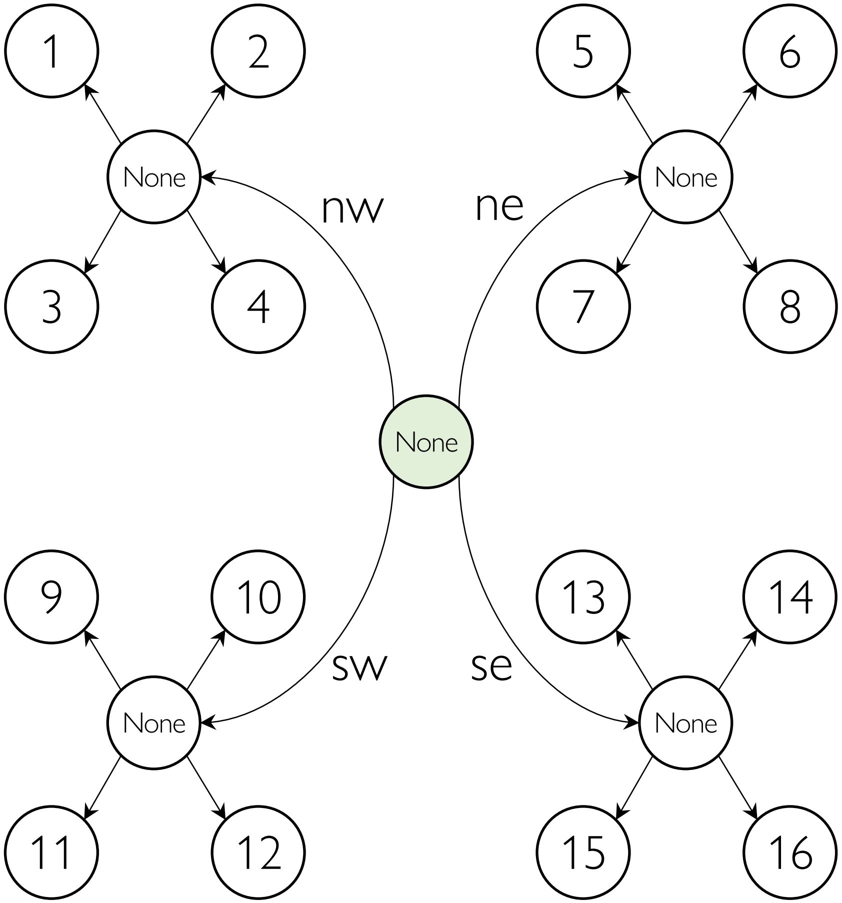


This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Quadtree to Image

- Traverse the quadtree and assign color to corresponding area with upleft corner at (ul_x, ul_y) and downright corner at (dr_x, dr_y)

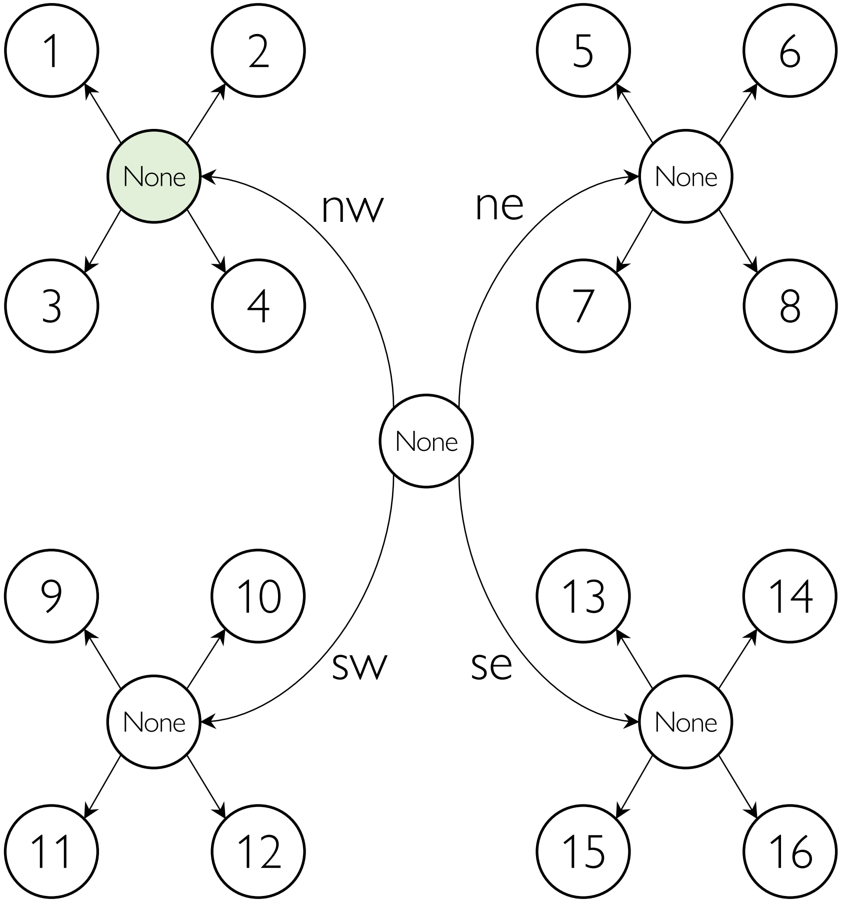




This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Quadtree to Image

- Traverse the quadtree and assign color to corresponding area with upleft corner at (ul_x, ul_y) and downright corner at (dr_x, dr_y)



column

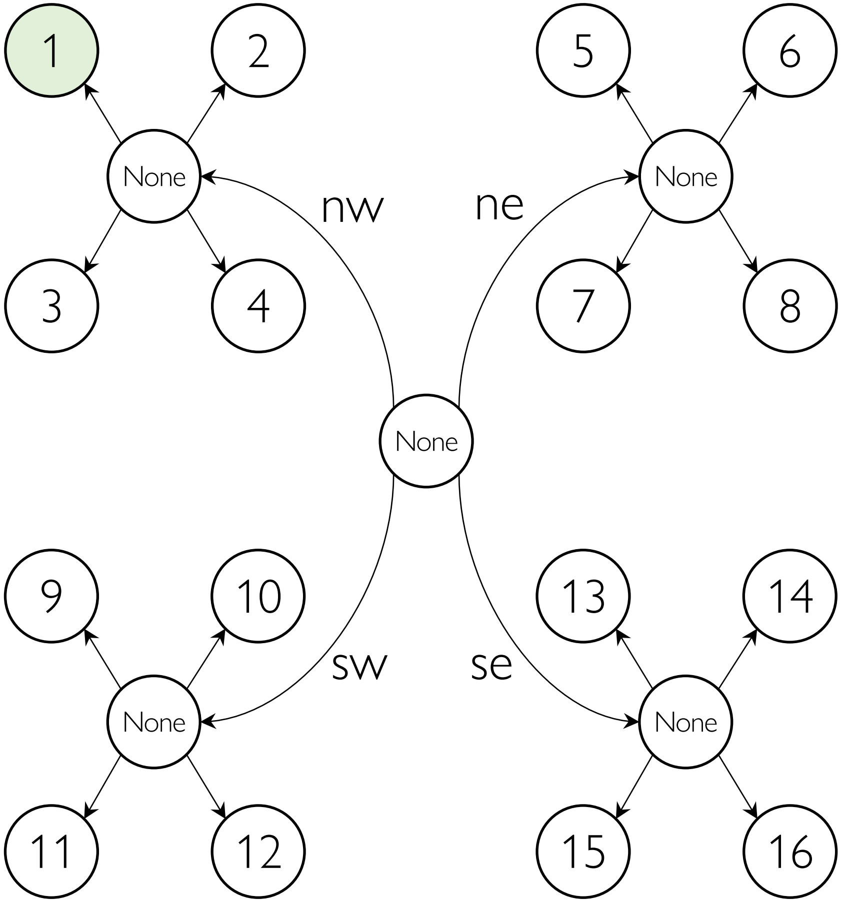
row

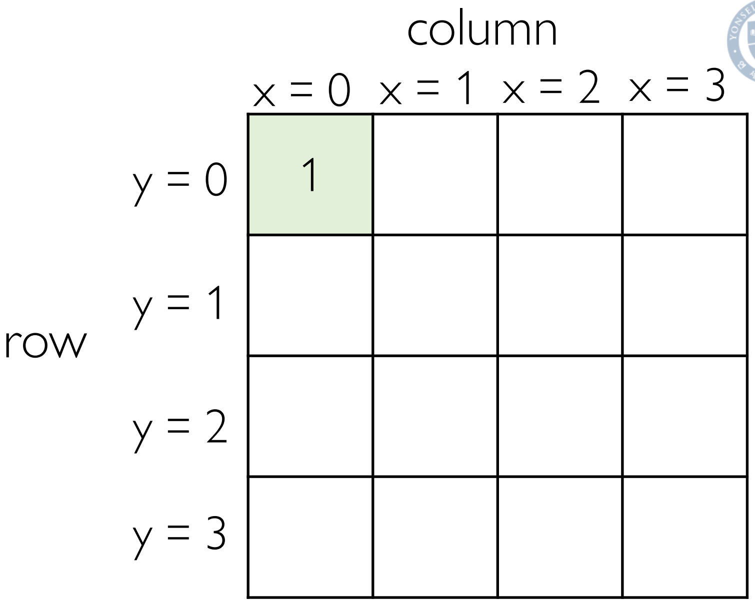
	x = 0	x = 1	x = 2	x = 3
y = 0				
y = 1				
y = 2				
y = 3				

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Quadtree to Image

- Traverse the quadtree and assign color to corresponding area with upleft corner at (ul_x, ul_y) and downright corner at (dr_x, dr_y)




column

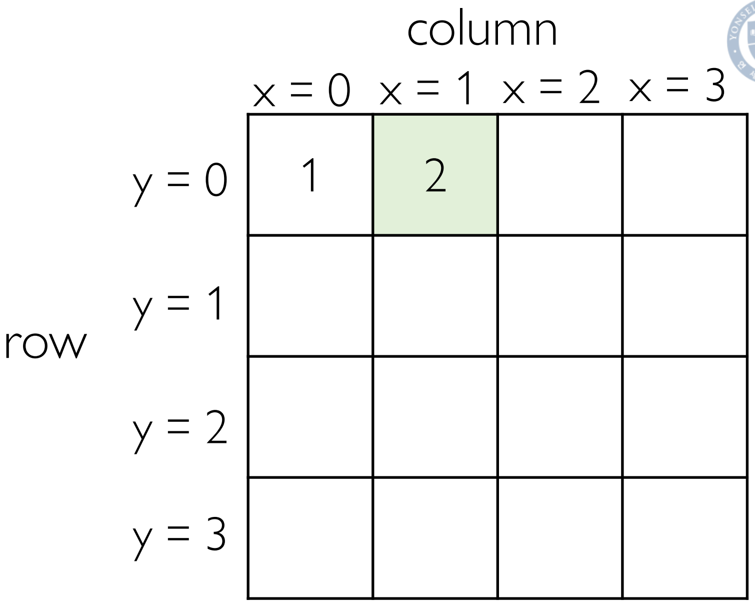
row

	x = 0	x = 1	x = 2	x = 3
y = 0	1			
y = 1				
y = 2				
y = 3				

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Quadtree to Image

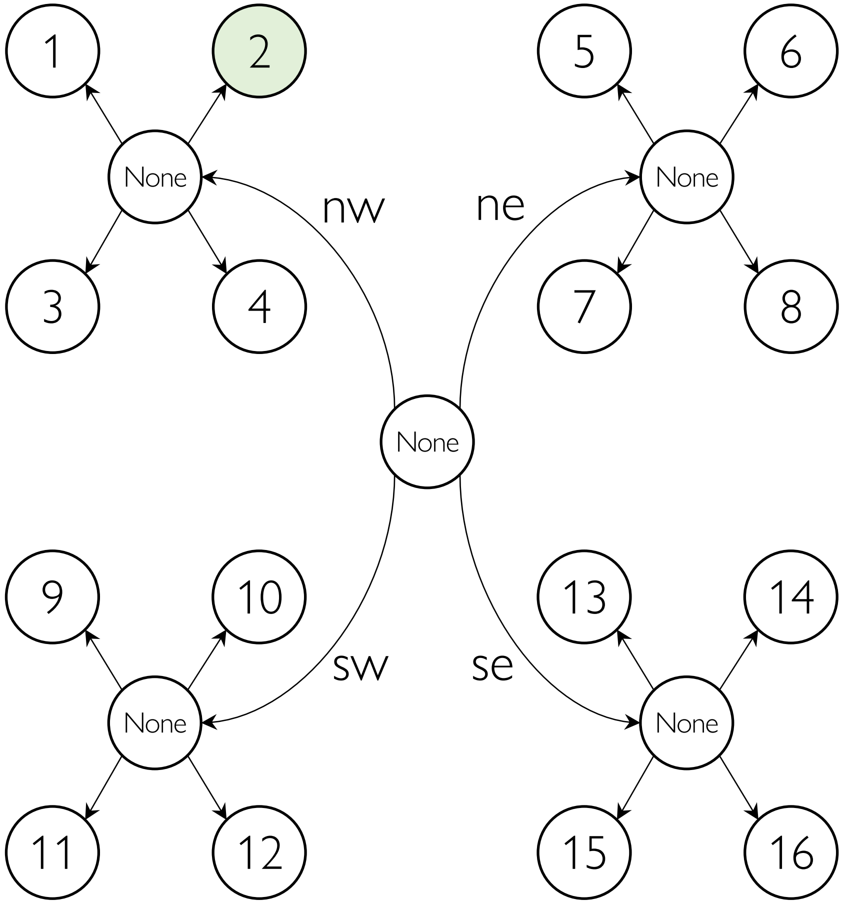
- Traverse the quadtree and assign color to corresponding area with upleft corner at (ul_x, ul_y) and downright corner at (dr_x, dr_y)



column

row

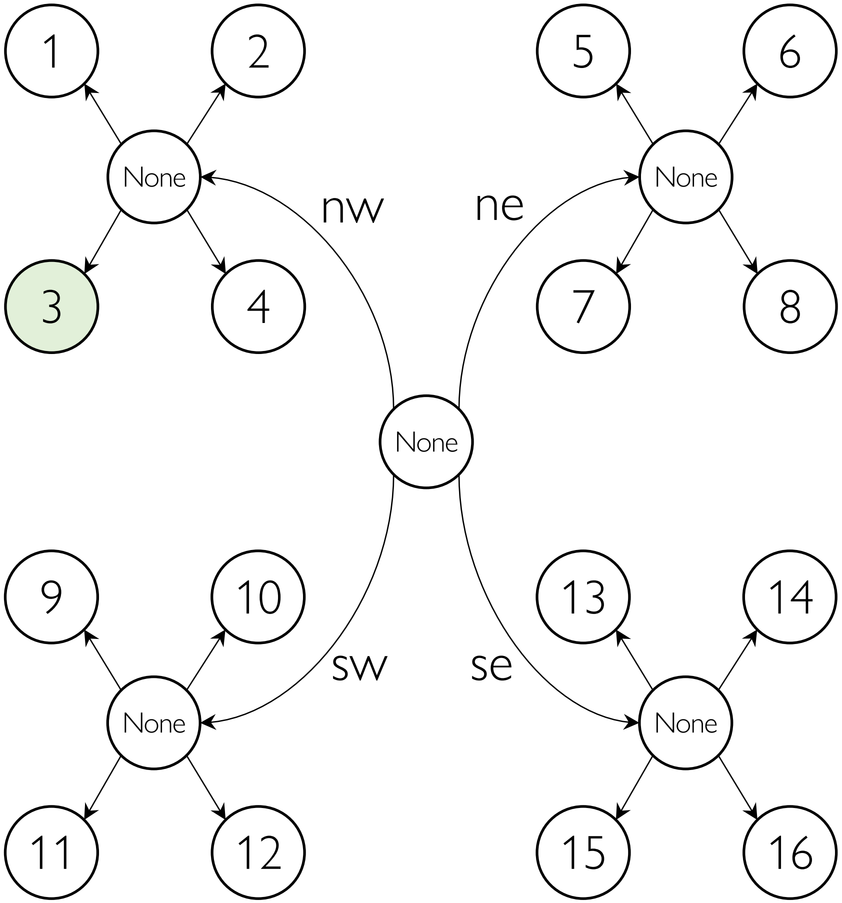
	x = 0	x = 1	x = 2	x = 3
y = 0	1	🟩 2		
y = 1				
y = 2				
y = 3				



This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Quadtree to Image

- Traverse the quadtree and assign color to corresponding area with upleft corner at (ul_x, ul_y) and downright corner at (dr_x, dr_y)



column

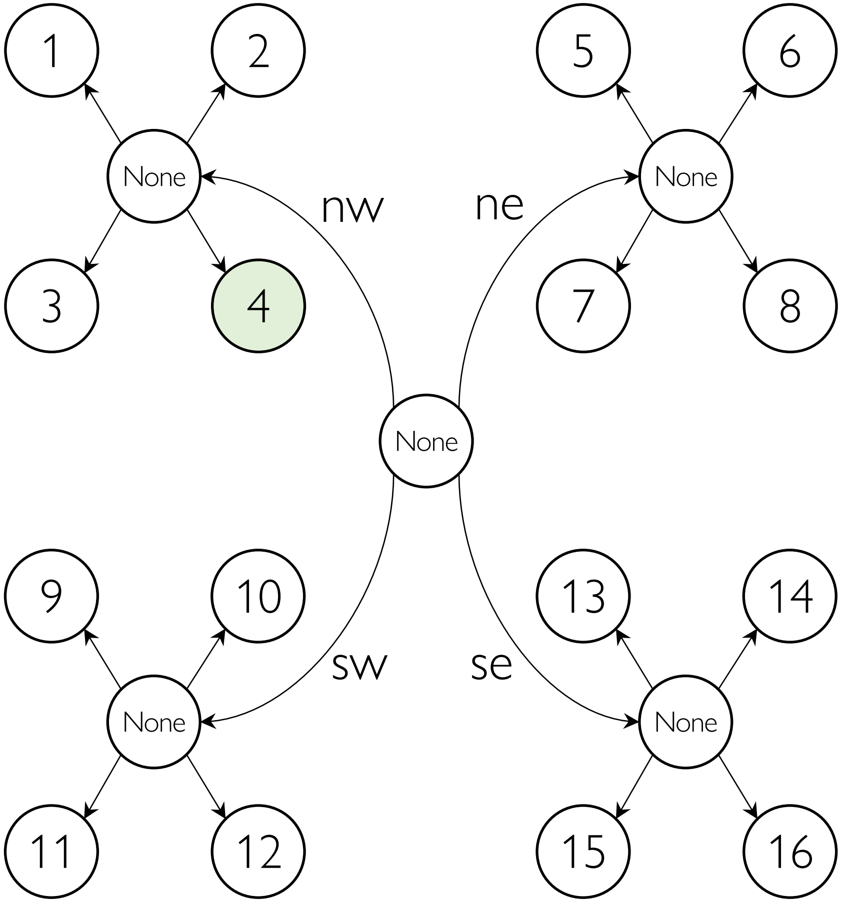
row

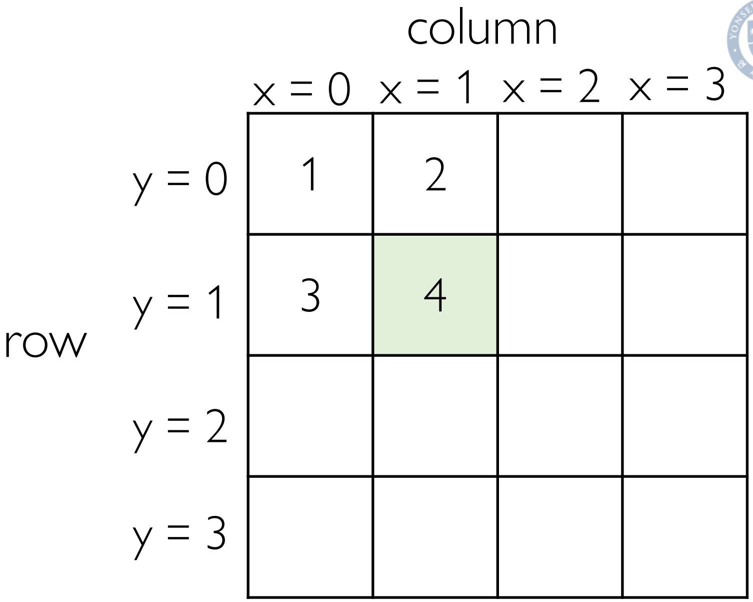
	x = 0	x = 1	x = 2	x = 3
y = 0	1	2		
y = 1	3			
y = 2				
y = 3				

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Quadtree to Image

- Traverse the quadtree and assign color to corresponding area with upleft corner at (ul_x, ul_y) and downright corner at (dr_x, dr_y)




column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	1	2		
y = 1	3	4		
y = 2				
y = 3				

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

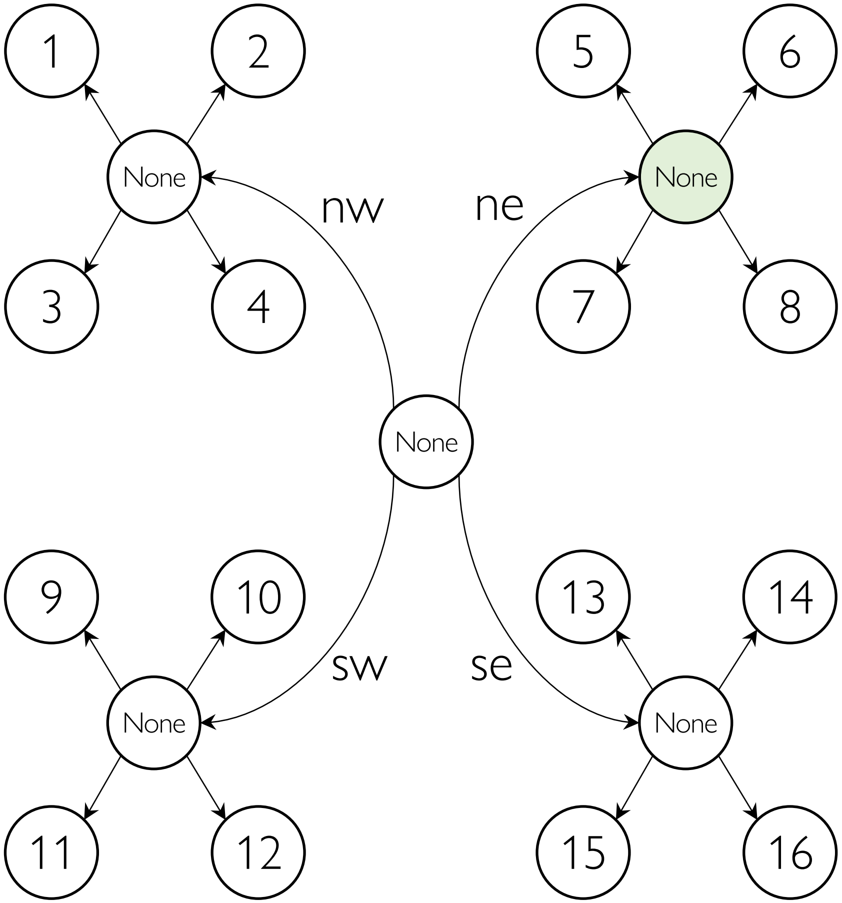
Quadtree to Image

- Traverse the quadtree and assign color to corresponding area with upleft corner at (ul_x, ul_y) and downright corner at (dr_x, dr_y)

column

row

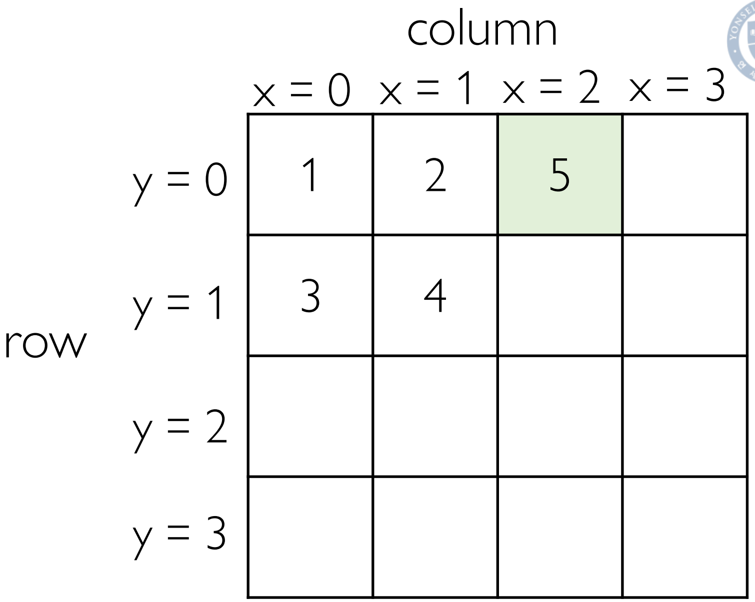
	x = 0	x = 1	x = 2	x = 3
y = 0	1	2		
y = 1	3	4		
y = 2				
y = 3				



This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Quadtree to Image

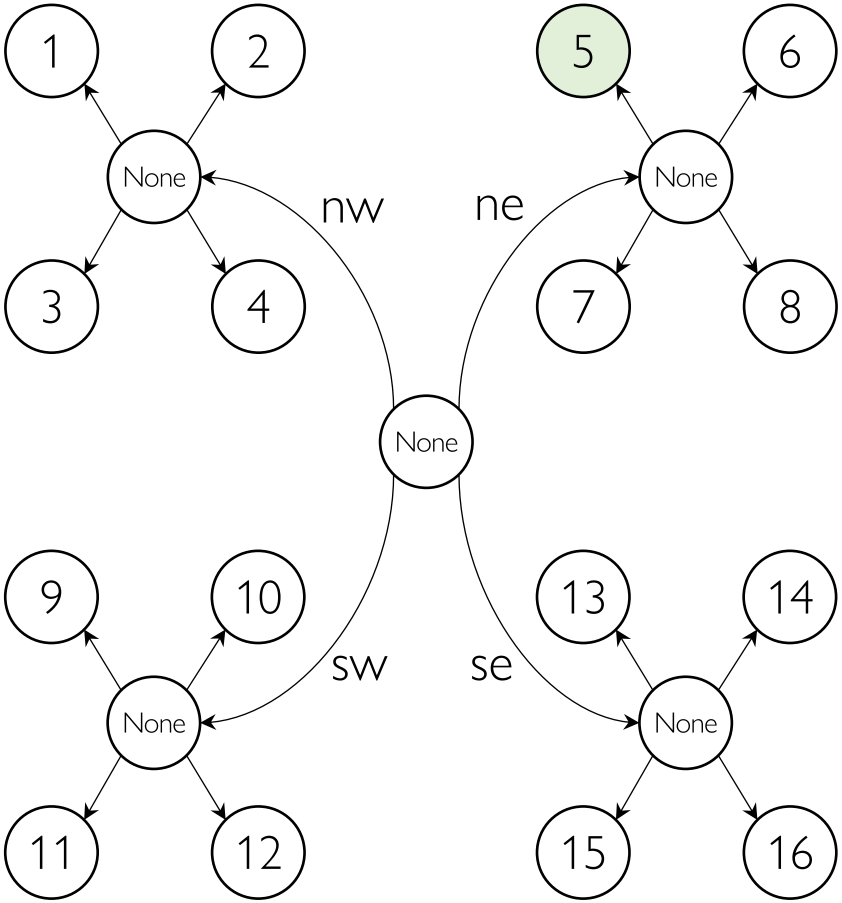
- Traverse the quadtree and assign color to corresponding area with upleft corner at (ul_x, ul_y) and downright corner at (dr_x, dr_y)



column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5	
y = 1	3	4		
y = 2				
y = 3				



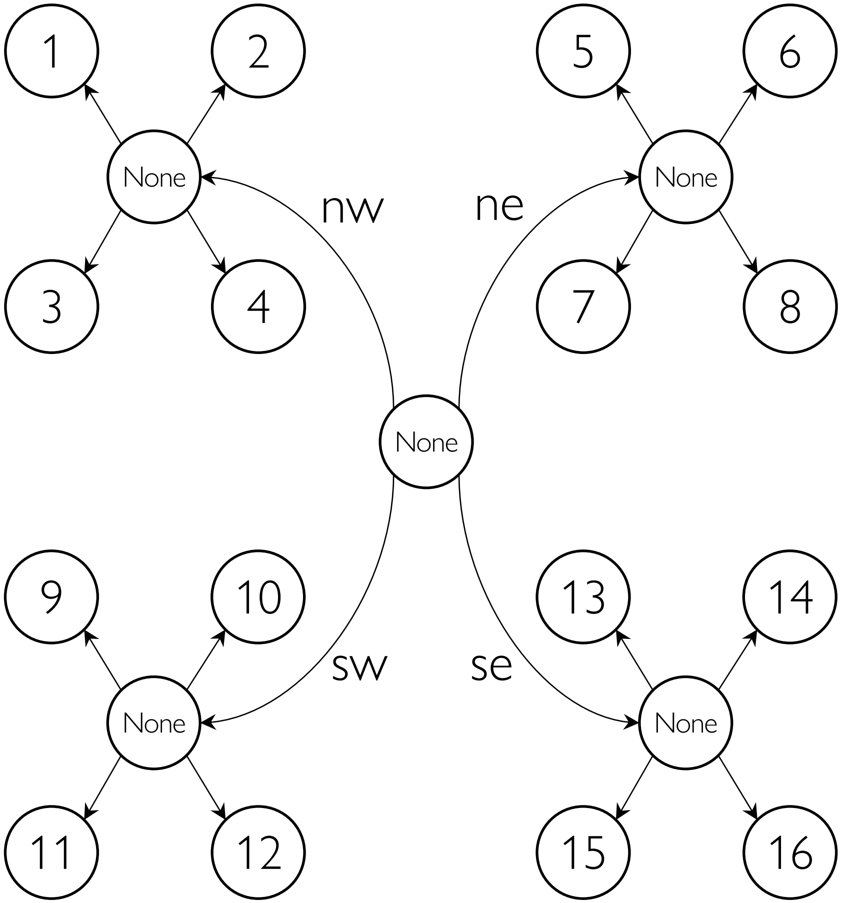
This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Quadtree to Image

- Traverse the quadtree and assign color to corresponding area with upleft corner at (ul_x, ul_y) and downright corner at (dr_x, dr_y)

column

row	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5	6
y = 1	3	4	7	8
y = 2	9	10	13	14
y = 3	11	12	15	16



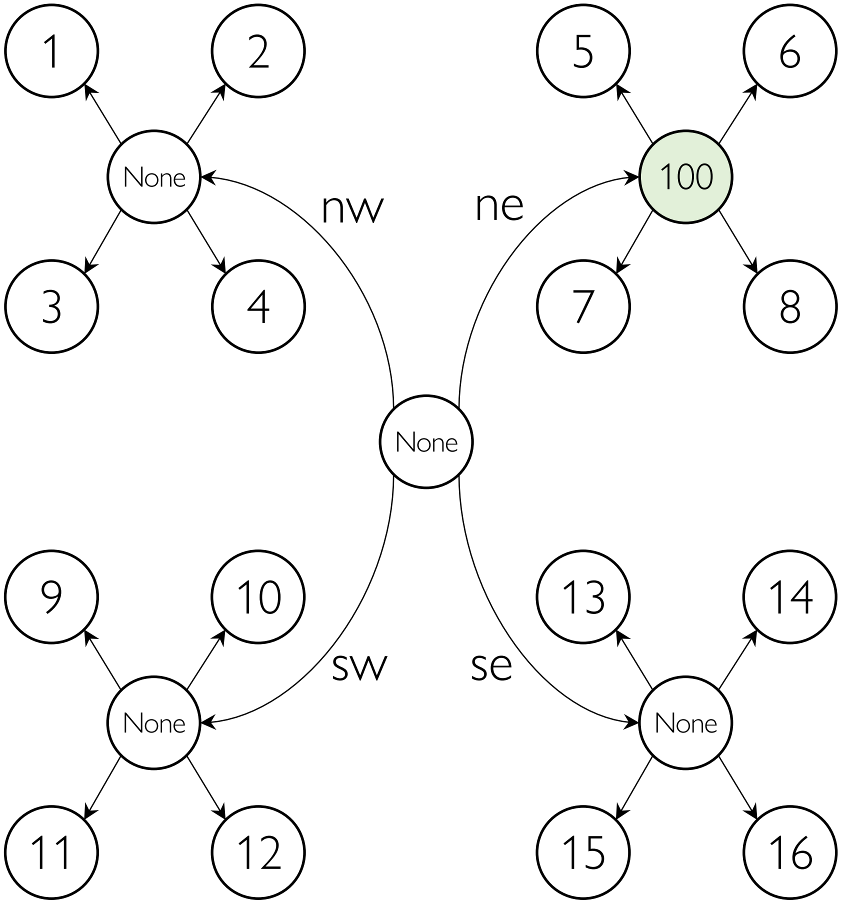
This is when we have colors at leaves!

What happens if we have colors at internal nodes?

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Quadtree to Image

- Traverse the quadtree and assign color to corresponding area with upleft corner at (ul_x, ul_y) and downright corner at (dr_x, dr_y)



column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5	6
y = 1	3	4	7	8
y = 2	9	10	13	14
y = 3	11	12	15	16

This is when we have colors at leaves!

What happens if we have colors at internal nodes?

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

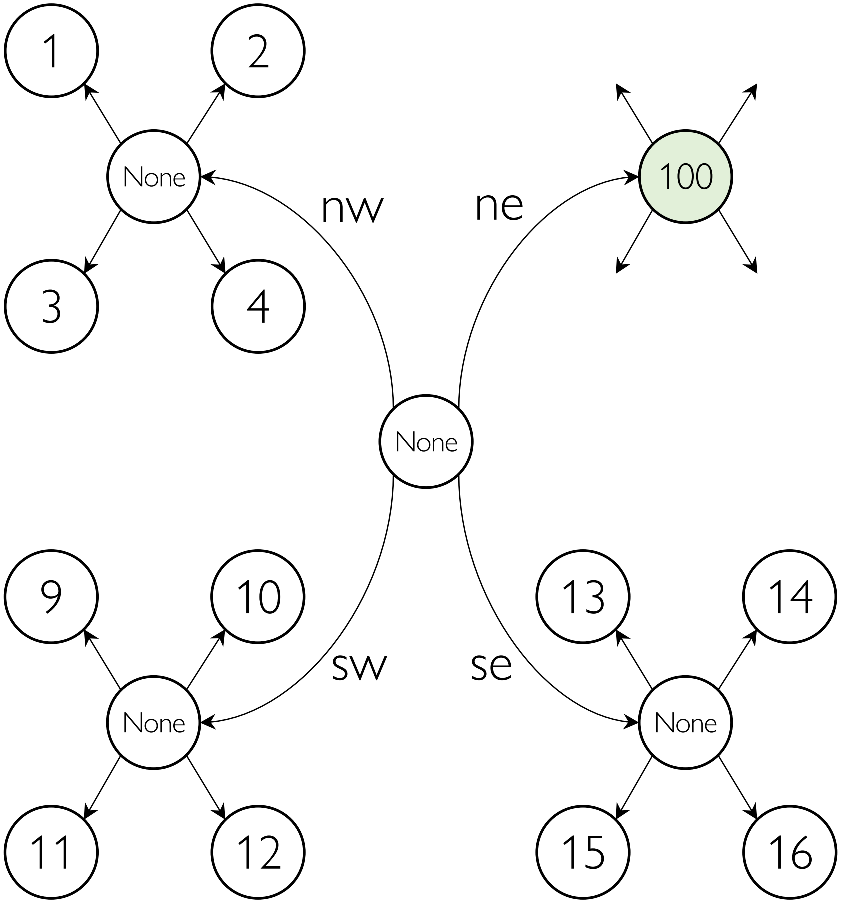
Quadtree to Image

- Traverse the quadtree and assign color to corresponding area with upleft corner at (ul_x, ul_y) and downright corner at (dr_x, dr_y)

column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	100	100
y = 1	3	4	100	100
y = 2	9	10	13	14
y = 3	11	12	15	16



This is when we have colors at leaves!

What happens if we have colors at internal nodes?

1. The internal node's entire area has that single value
2. Its children do not exist
    1. Removal happens in Part 1C
    2. We do NOT need to traverse its children anyway

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

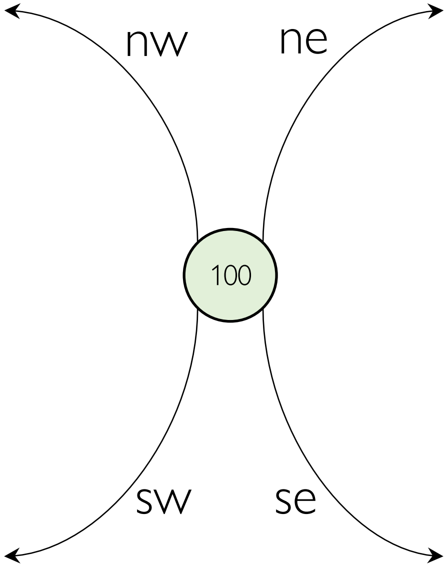
Quadtree to Image

- Traverse the quadtree and assign color to corresponding area with upleft corner at (ul_x, ul_y) and downright corner at (dr_x, dr_y)

column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	100	100	100	100
y = 1	100	100	100	100
y = 2	100	100	100	100
y = 3	100	100	100	100



One extreme case

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

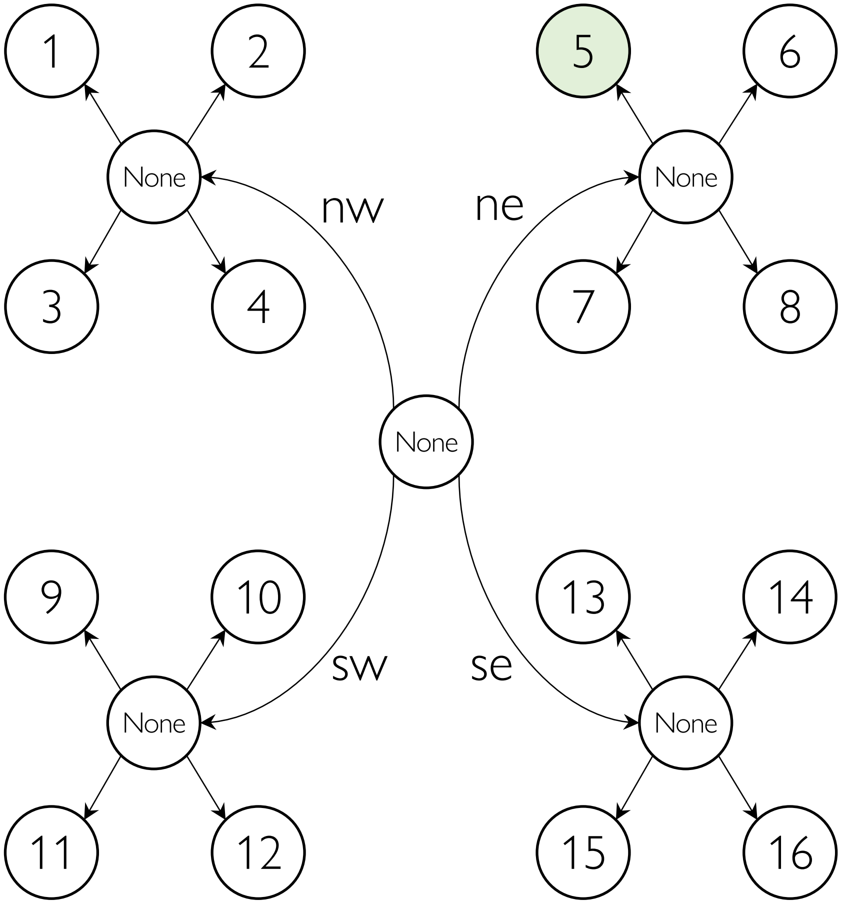
Quadtree to Image

- Traverse the quadtree and assign color to corresponding area with upleft corner at (ul_x, ul_y) and downright corner at (dr_x, dr_y)

column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	🟩 5	6
y = 1	3	4	7	8
y = 2	9	10	13	14
y = 3	11	12	15	16



Notice that our logic does not change:

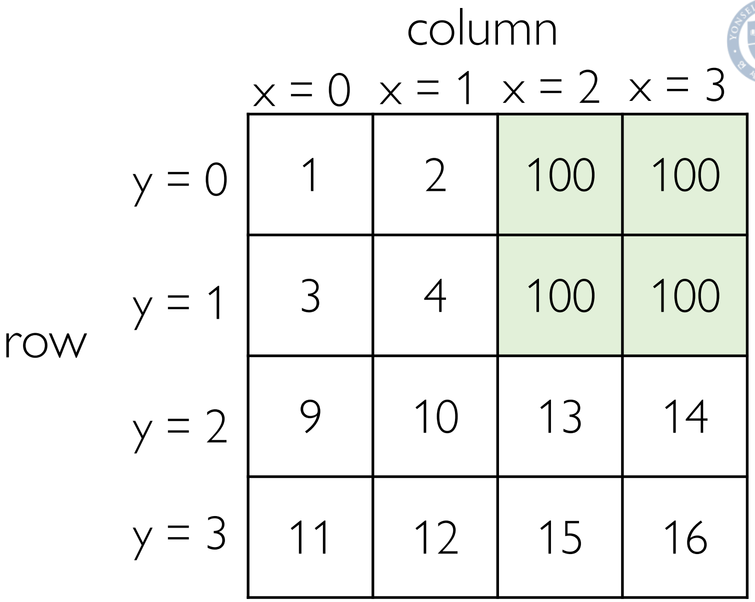
1. for a leaf node, its "area" is just a single pixel
2. for an internal node, its "area" is just some area larger than a pixel

The "areas" are both defined by (ul_x, ul_y) and (dr_x, dr_y)

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

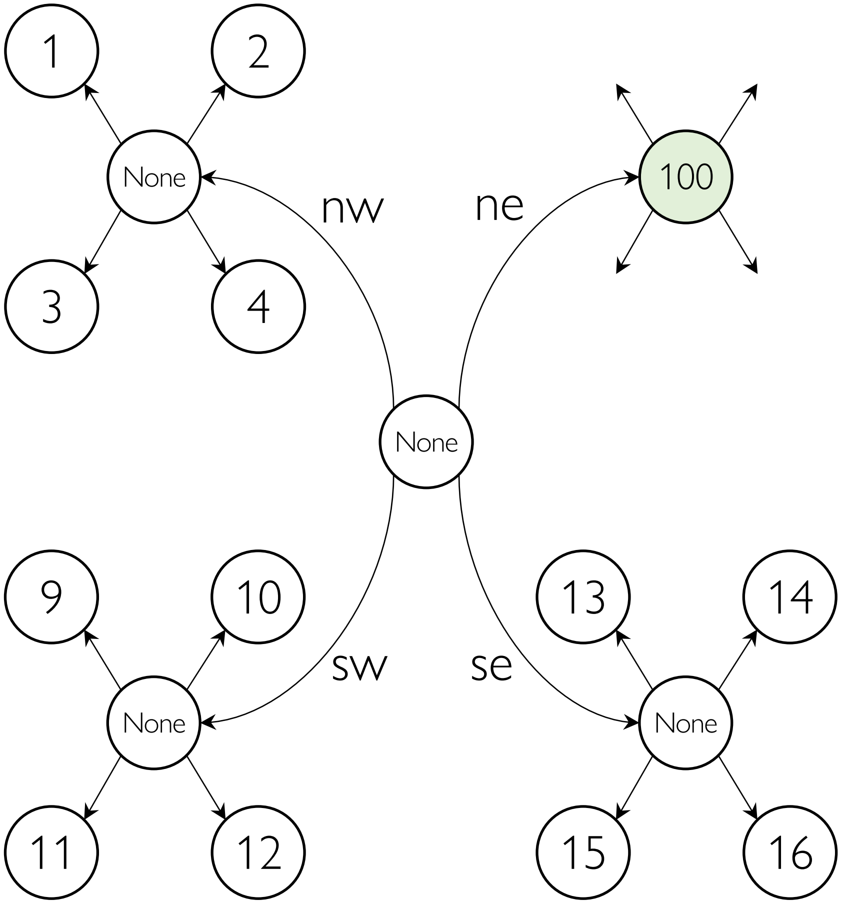
Quadtree to Image

- Traverse the quadtree and assign color to corresponding area with upleft corner at (ul_x, ul_y) and downright corner at (dr_x, dr_y)



column

row	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	100	100
y = 1	3	4	100	100
y = 2	9	10	13	14
y = 3	11	12	15	16



Notice that our logic does not change:

1. for a leaf node, its "area" is just a single pixel
2. for an internal node, its "area" is just some area larger than a pixel

The "areas" are both defined by (ul_x, ul_y) and (dr_x, dr_y)

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Quadtree to Image

- Implement quadtree_to_image_recursive
- Recursively pass down 'qt' as argument to modify its image 'qt.image' throughout the recursion
  - This is an example of a preorder traversal: you check the condition of the parent node before deciding to traverse its children
- Use draw_color(qt, node, draw_box) to directly modify qt.image
  - draw_box argument used for Part 2 to draw boxes on the image for visualization of "areas"
- Again, not a very long code

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Part 1C: Compression: How to set colors for internal nodes?

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

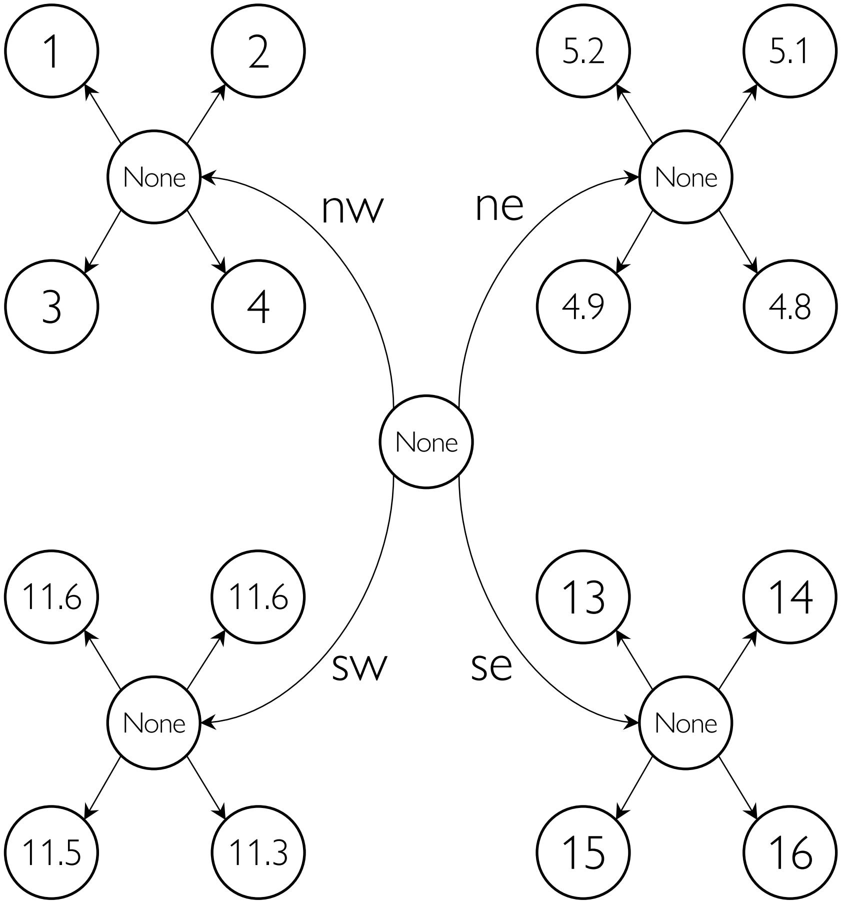
Compression

- Suppose we have a quadtree from the image shown at the right

column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5.2	5.1
y = 1	3	4	4.9	4.8
y = 2	11.6	11.6	13	14
y = 3	11.5	11.3	15	16



We want to "compress" the image by simplifying pixels that are

1. siblings and
2. have "similar" color values

If an internal node's children have "similar" color values, then

1. the internal node's color = average of the children colors
2. children nodes are removed (pruned)

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

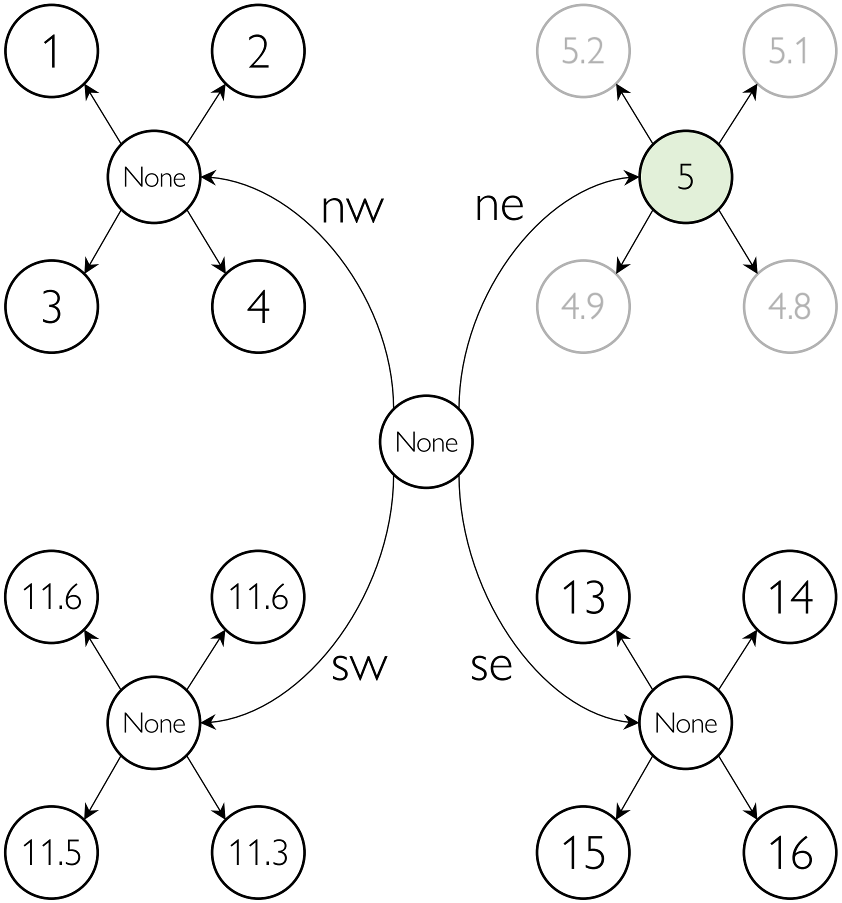
Compression

- Suppose we have a quadtree from the image shown at the right

column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5.2	5.1
y = 1	3	4	4.9	4.8
y = 2	11.6	11.6	13	14
y = 3	11.5	11.3	15	16



We want to "compress" the image by simplifying pixels that are

1. siblings and
2. have "similar" color values

If an internal node's children have "similar" color values, then

1. the internal node's color = average of the children colors
2. children nodes are removed (pruned)

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

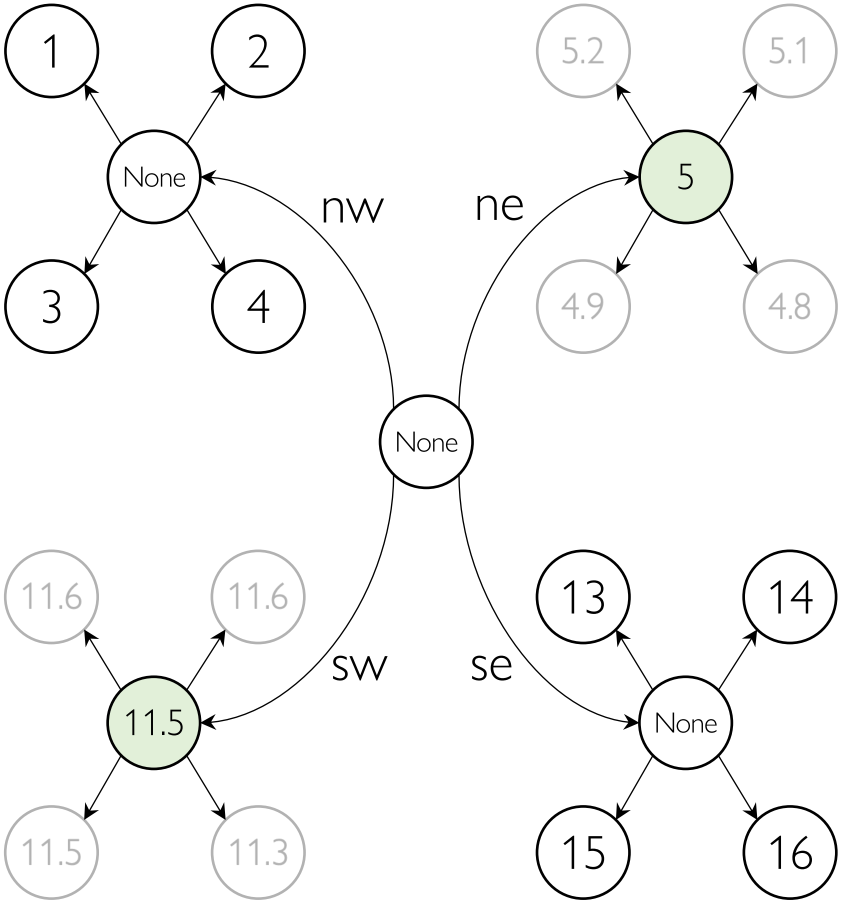
Compression

- Suppose we have a quadtree from the image shown at the right

column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5.2	5.1
y = 1	3	4	4.9	4.8
y = 2	11.6	11.6	13	14
y = 3	11.5	11.3	15	16



We want to "compress" the image by simplifying pixels that are

1. siblings and
2. have "similar" color values

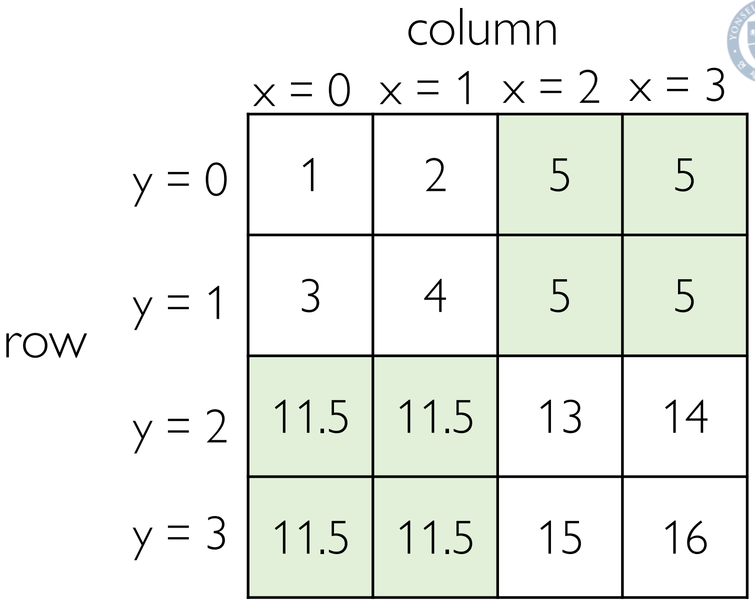
If an internal node's children have "similar" color values, then

1. the internal node's color = average of the children colors
2. children nodes are removed (pruned)

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

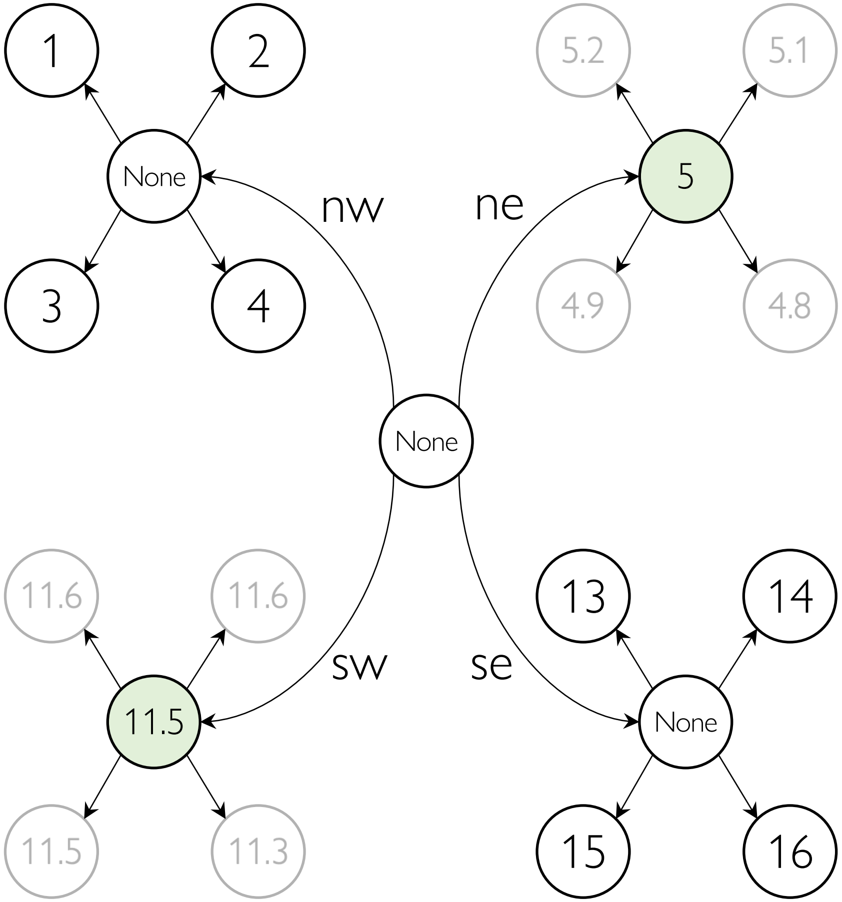
Compression

- Suppose we have a quadtree from the image shown at the right



column

		x = 0	x = 1	x = 2	x = 3
row	y = 0	1	2	5	5
	y = 1	3	4	5	5
	y = 2	11.5	11.5	13	14
	y = 3	11.5	11.5	15	16



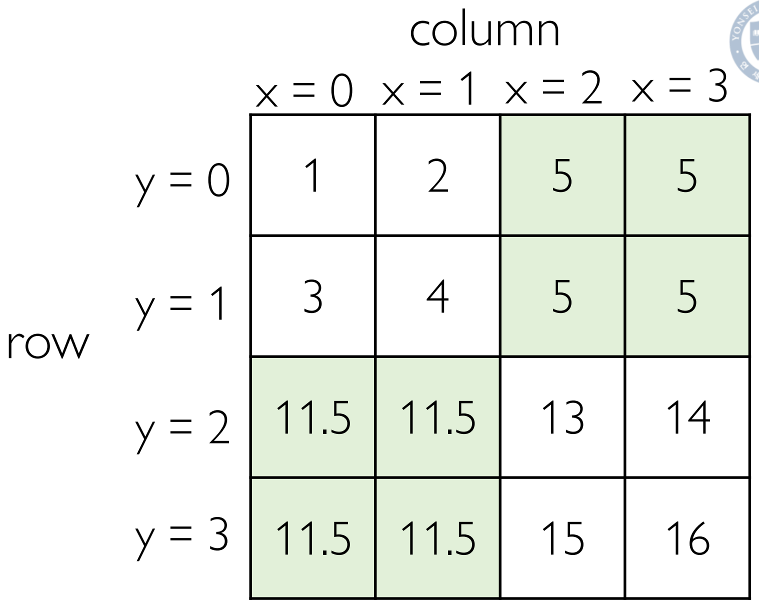
quadtree_to_image then returns the compressed image:

1. similar image areas are "compressed" to one (most representative) color
   1. this reduces the image file size
2. reduces the # of trees of quadtree

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Compression

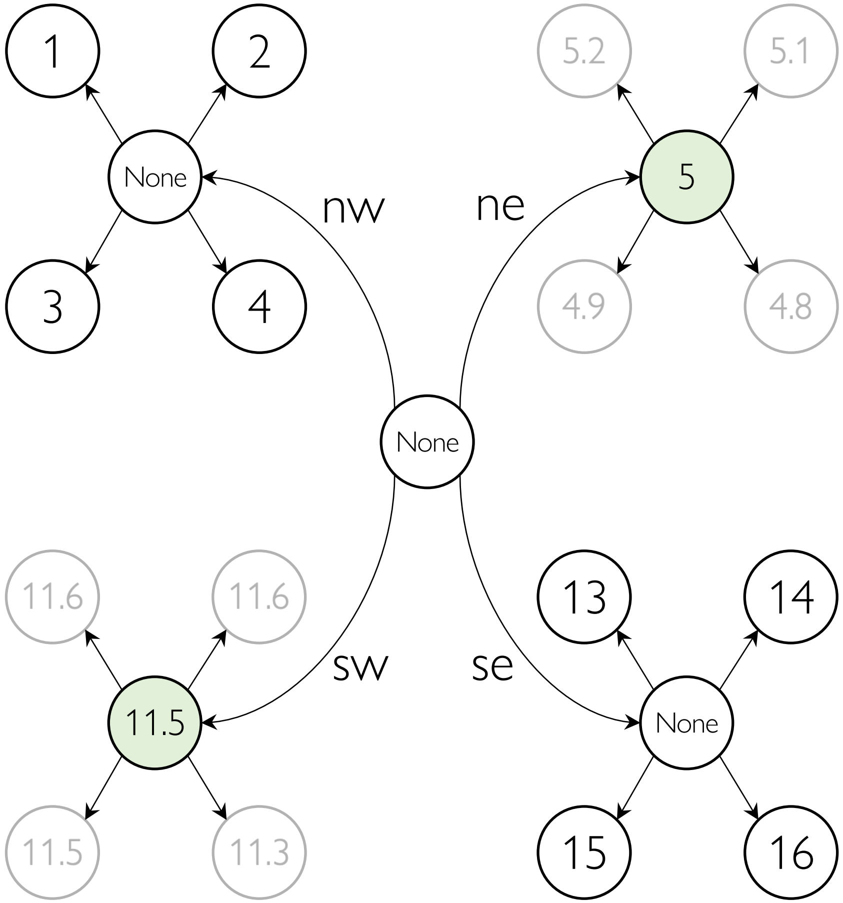
- Suppose we have a quadtree from the image shown at the right



column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5	5
y = 1	3	4	5	5
y = 2	11.5	11.5	13	14
y = 3	11.5	11.5	15	16



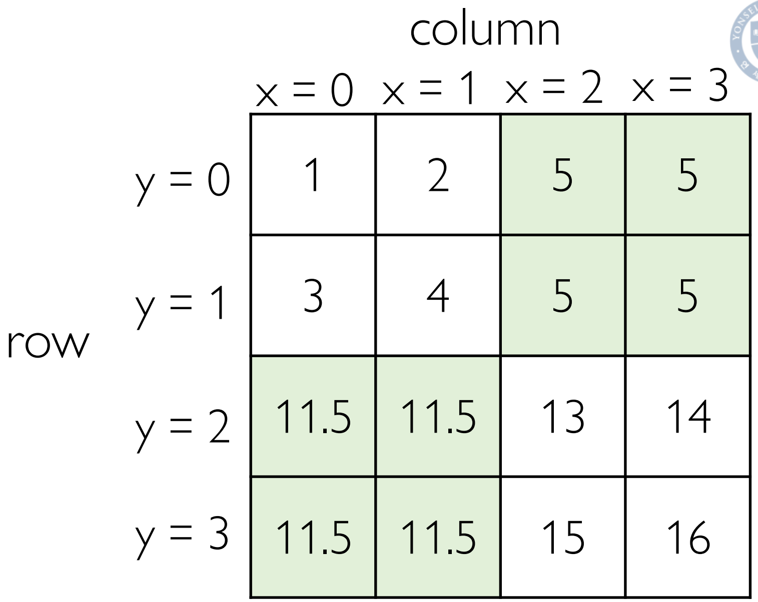
use combine_colors(node, threshold) to update the internal node color

1. It checks for similarity by computing the “difference” among the children colors
2. If the “difference” is less than some “standard” we call “threshold” (difference < threshold)
    1. The similarity is high, so combine_colors returns the average children color
3. If the difference >= threshold
    1. The similarity is small, so combine_colors returns None

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

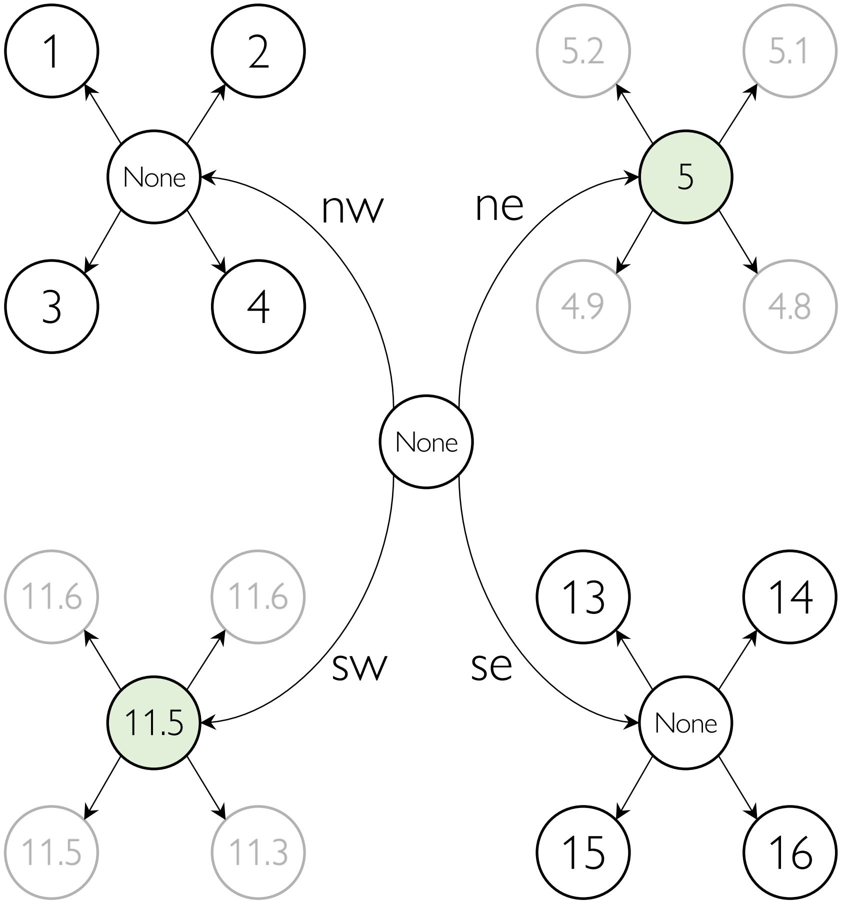
Compression

- Suppose we have a quadtree from the image shown at the right



column

row	x = 0	x = 1	x = 2	x = 3
y = 0	1	2	5	5
y = 1	3	4	5	5
y = 2	11.5	11.5	13	14
y = 3	11.5	11.5	15	16



This means your choice "threshold" determines the strength of compression. This is our choice.

1. Small threshold means you only combine children with very small difference
    1. weak, conservative compression
2. Large threshold means you are willing to combine children with larger difference
    1. strong, liberal compression

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

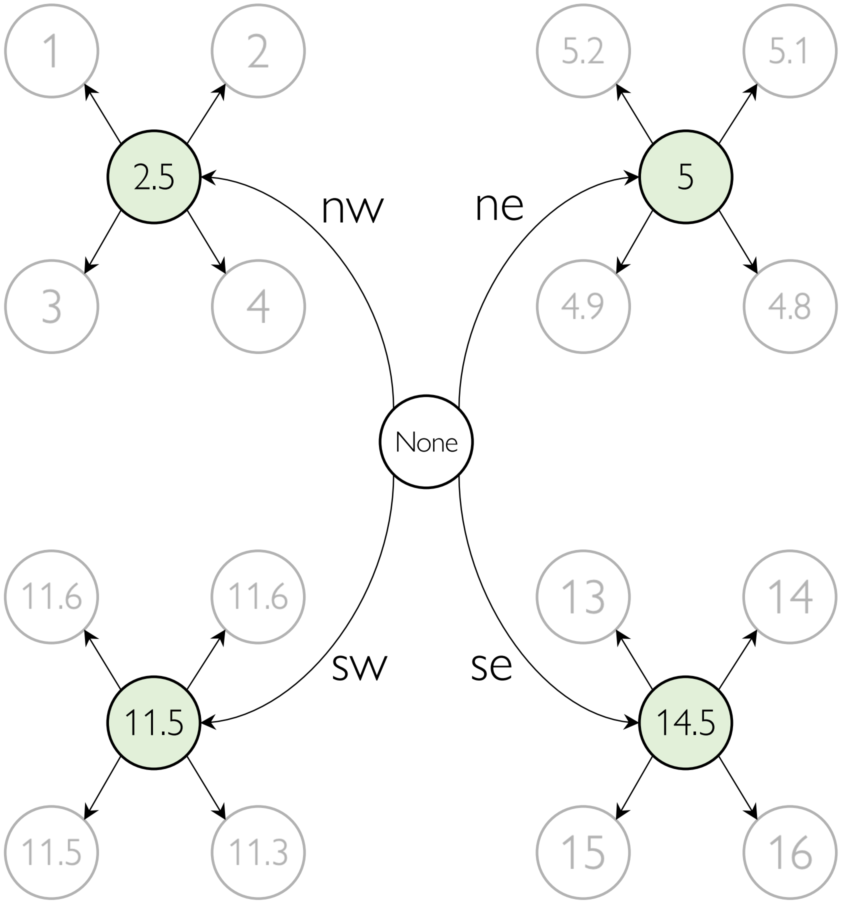
Compression

- Suppose we have a quadtree from the image shown at the right

column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	2.5	2.5	5	5
y = 1	2.5	2.5	5	5
y = 2	11.5	11.5	14.5	14.5
y = 3	11.5	11.5	14.5	14.5



This means your choice "threshold" determines the strength of compression. This is our choice.

1. Small threshold means you only combine children with very small difference
    1. weak, conservative compression
2. Large threshold means you are willing to combine children with larger difference
    1. strong, liberal compression

Ex: very large threshold may combine all children

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

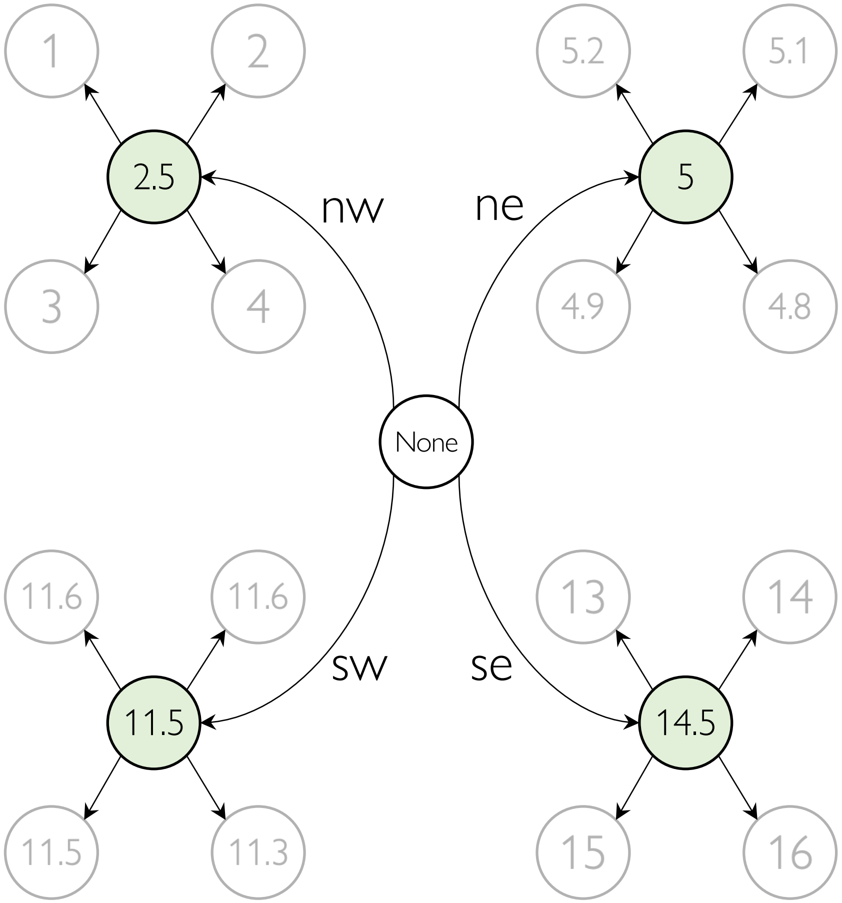
Compression

- Suppose we have a quadtree from the image shown at the right

column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	2.5	2.5	5	5
y = 1	2.5	2.5	5	5
y = 2	11.5	11.5	14.5	14.5
y = 3	11.5	11.5	14.5	14.5



Note that this can happen recursively from the bottom of quadtree!

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

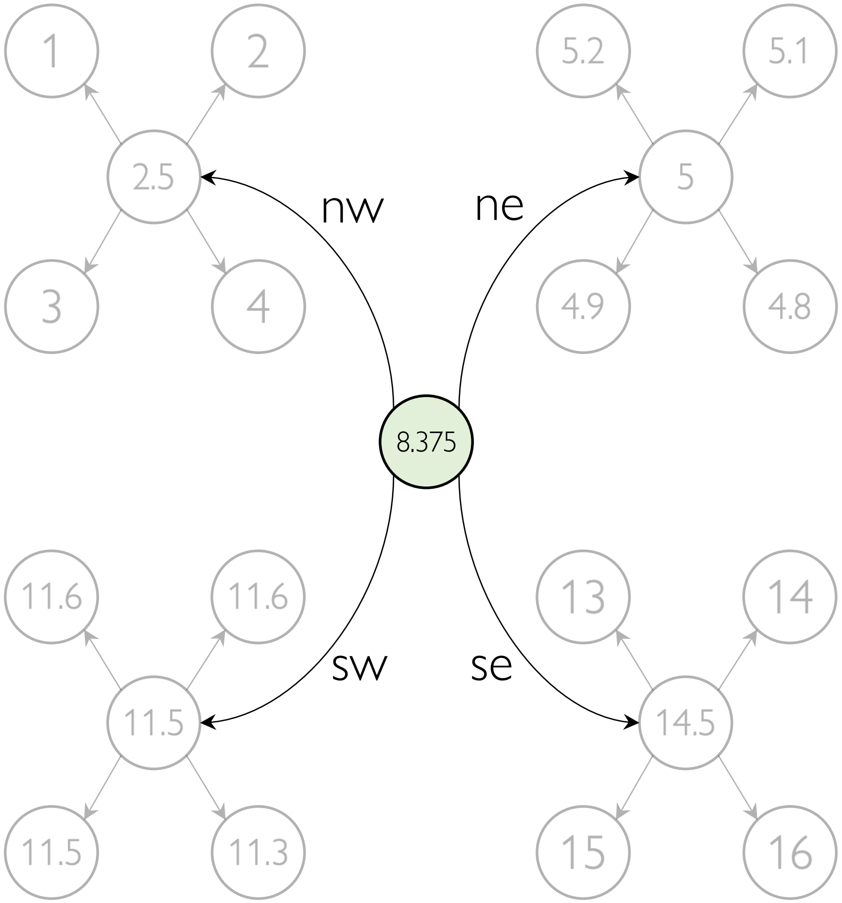
Compression

- Suppose we have a quadtree from the image shown at the right

column

row

	x = 0	x = 1	x = 2	x = 3
y = 0	8.375	8.375	8.375	8.375
y = 1	8.375	8.375	8.375	8.375
y = 2	8.375	8.375	8.375	8.375
y = 3	8.375	8.375	8.375	8.375



Note that this can happen recursively from the bottom of quadtree!

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

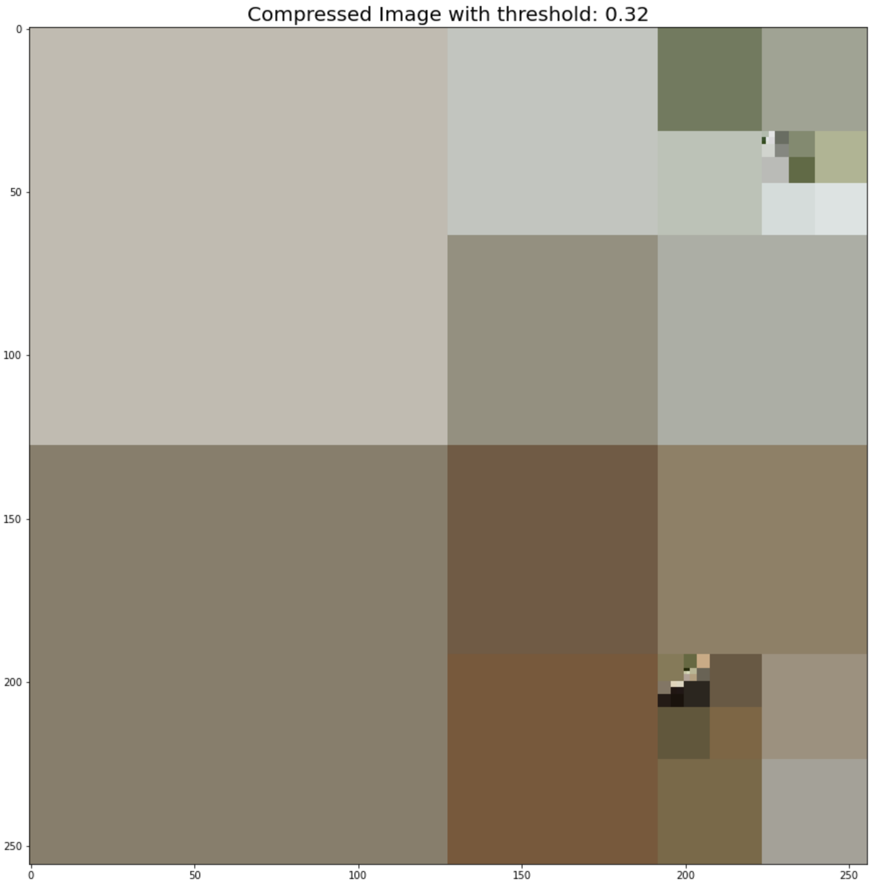
Compression

- Implement compress_quadtree_recursive
- Recursively pass down ‘qt’ as argument to update the internal node colors from the bottom of the quadtree
  - This is an example of a postorder traversal: you update the children colors by visiting them first
- Use combine_colors(node, threshold)
- Make sure to remove children if an internal node has a color
  - Easy to check if this happened by using count_nodes function you implement in Part1A
- assignment3.ipynb tries specific thresholds
  - I will grade with my choice of thresholds
- Again, not a very long code

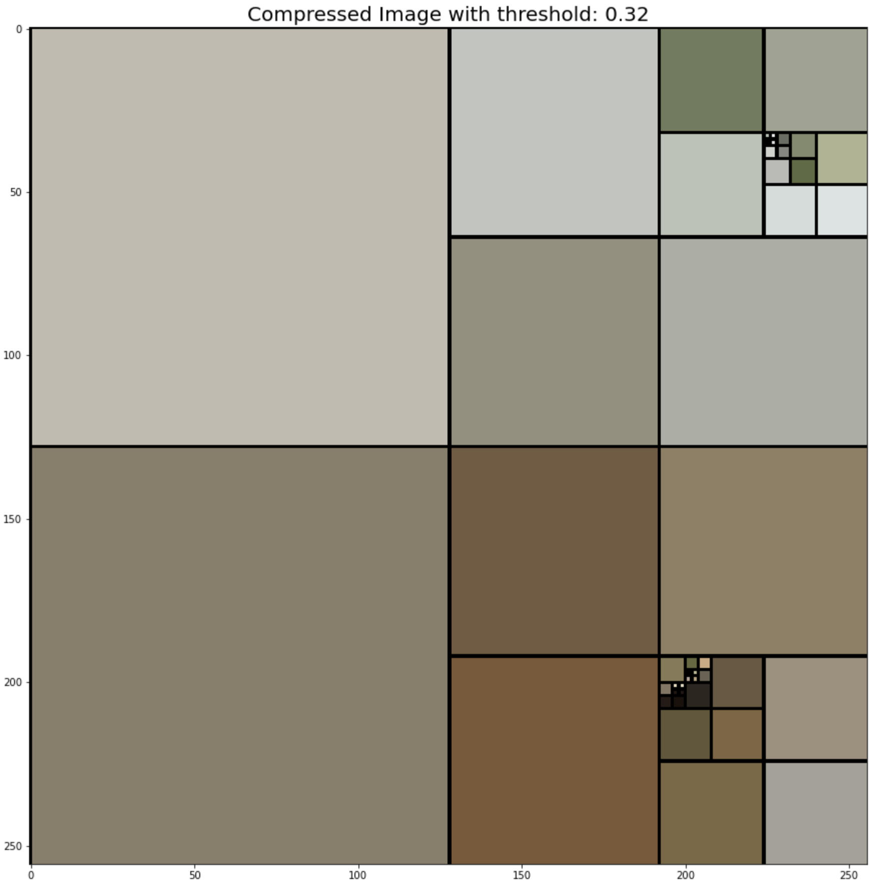
This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Part 2: Test on a real image

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

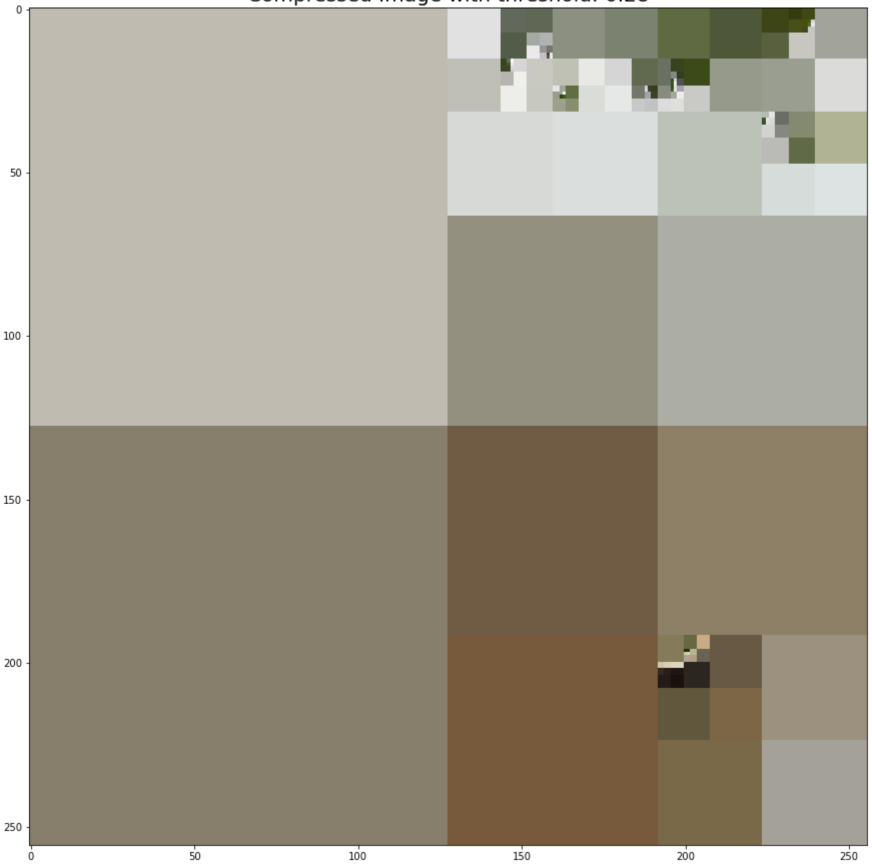


Compressed Image with threshold: 0.32

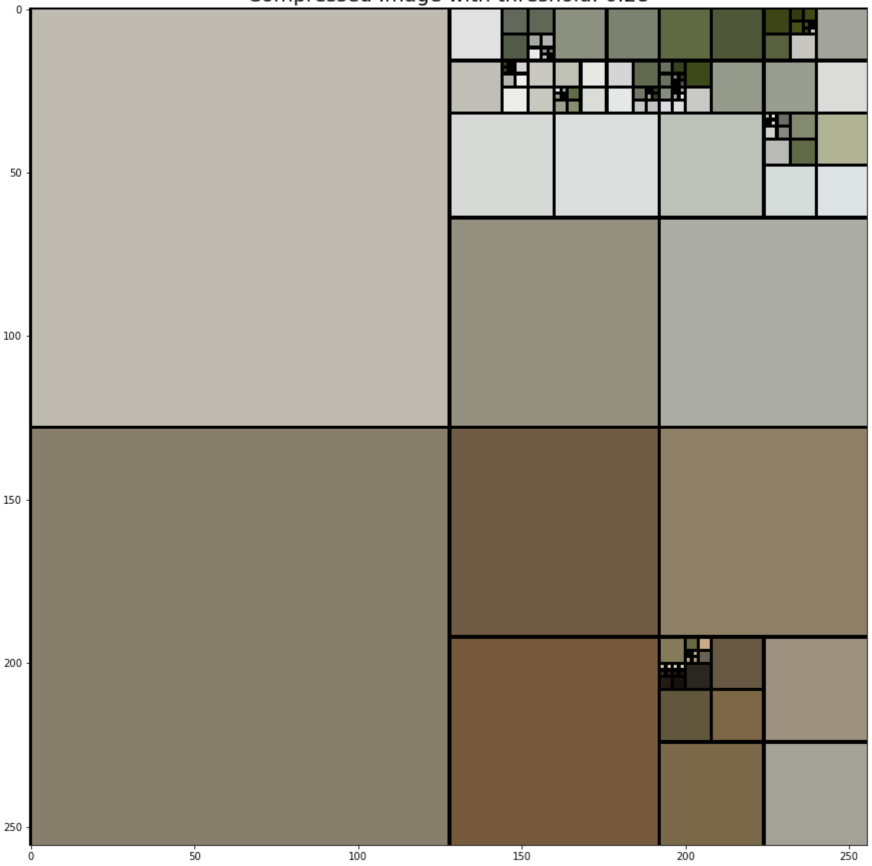


Compressed Image with threshold: 0.32

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.



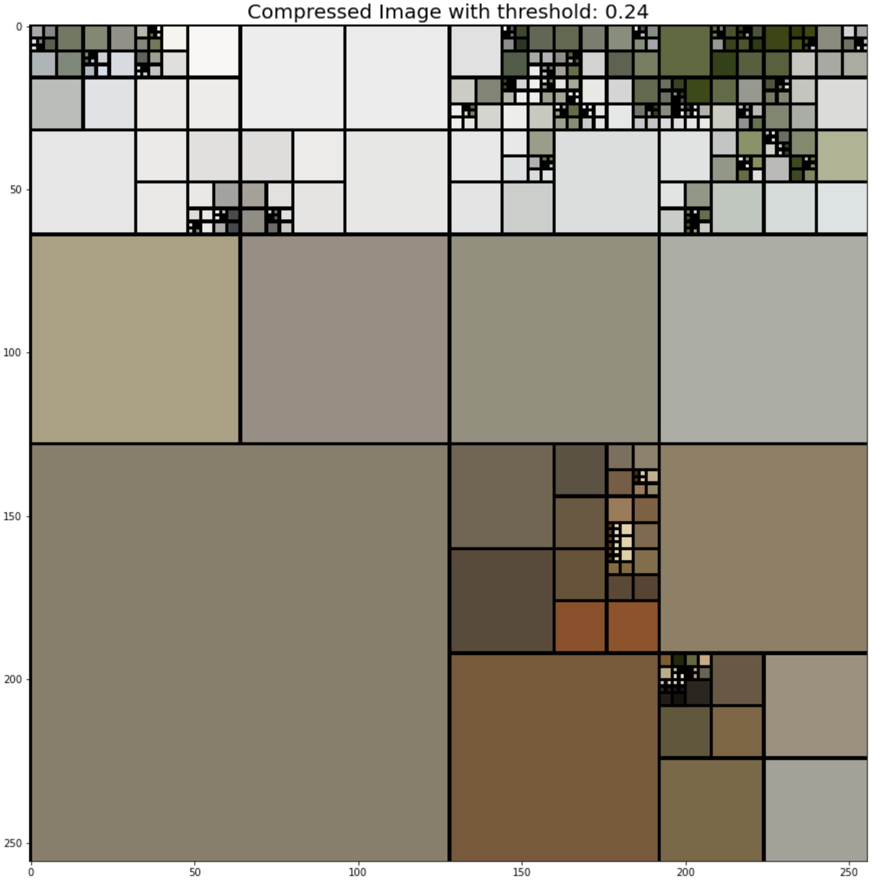
Compressed Image with threshold: 0.28



Compressed Image with threshold: 0.28

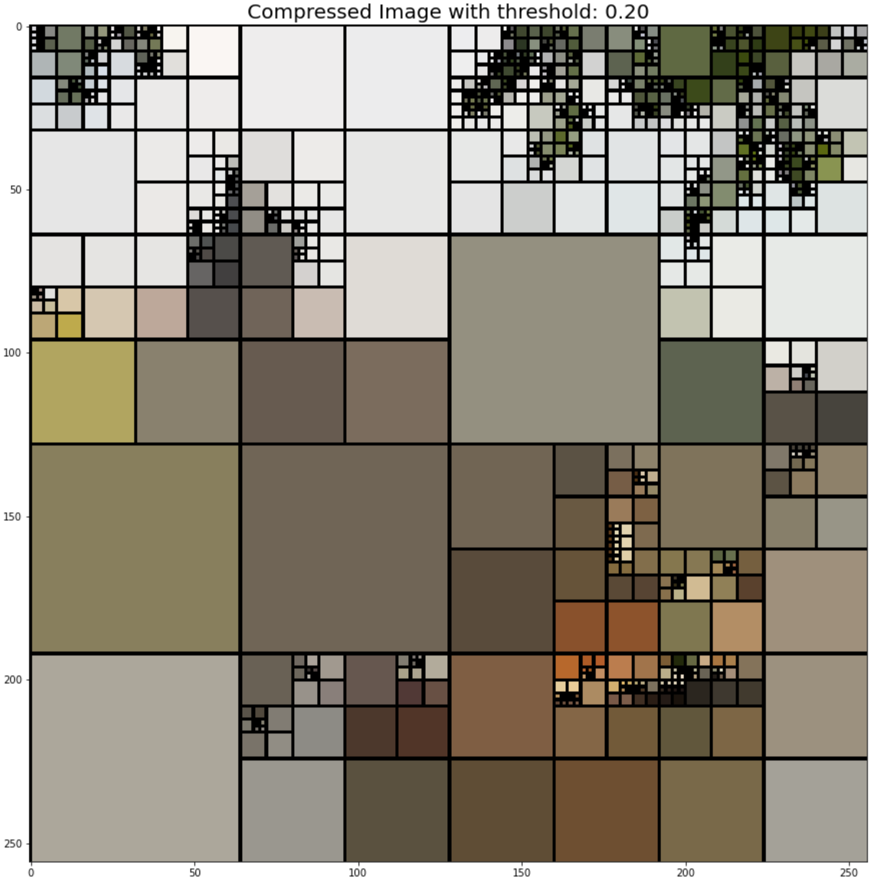
This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.


Compressed Image with threshold: 0.24


Compressed Image with threshold: 0.24

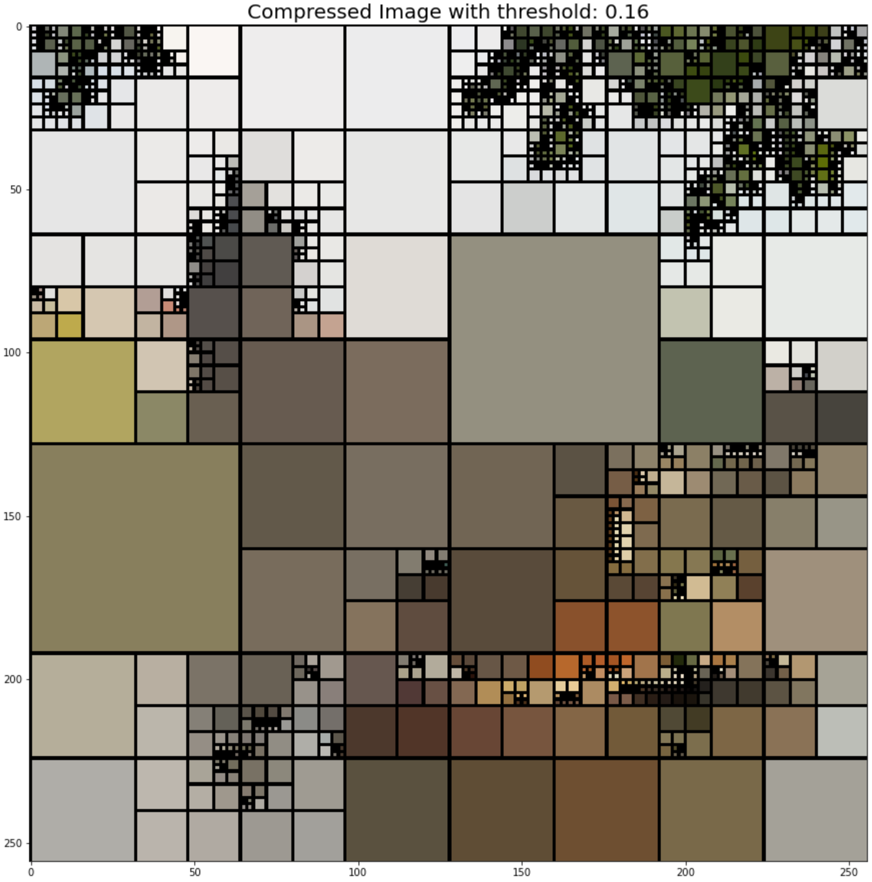
This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Compressed Image with threshold: 0.20

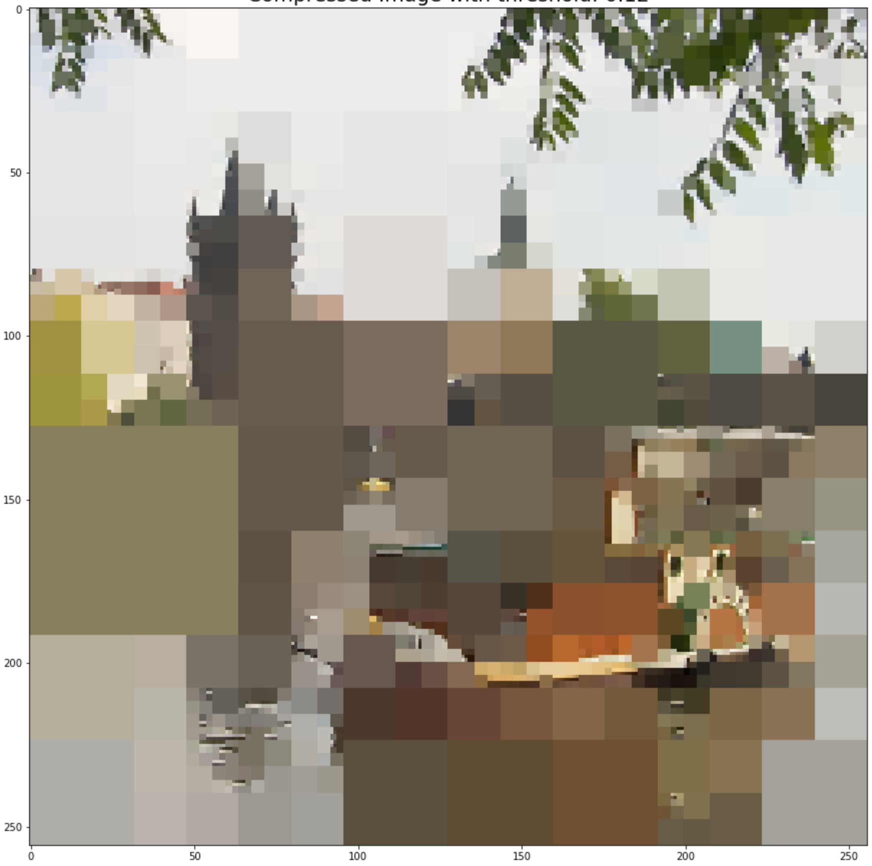


Compressed Image with threshold: 0.20

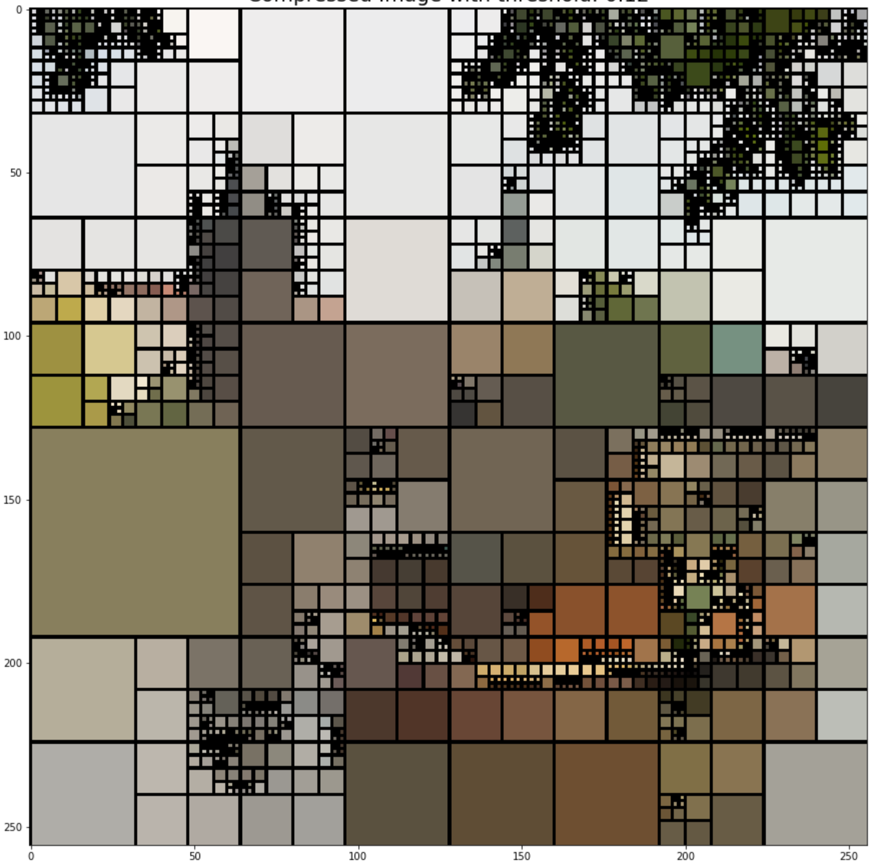
This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.



This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.



Compressed Image with threshold: 0.12



Compressed Image with threshold: 0.12

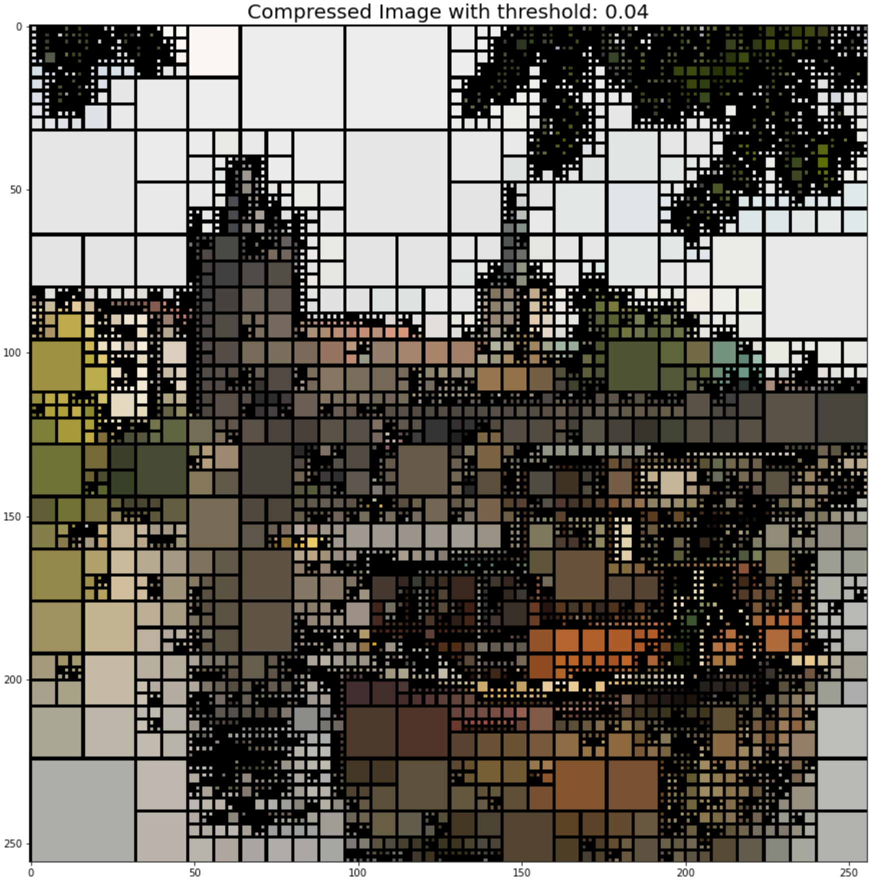
This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Compressed Image with threshold: 0.08

Compressed Image with threshold: 0.08

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Compressed Image with threshold: 0.04



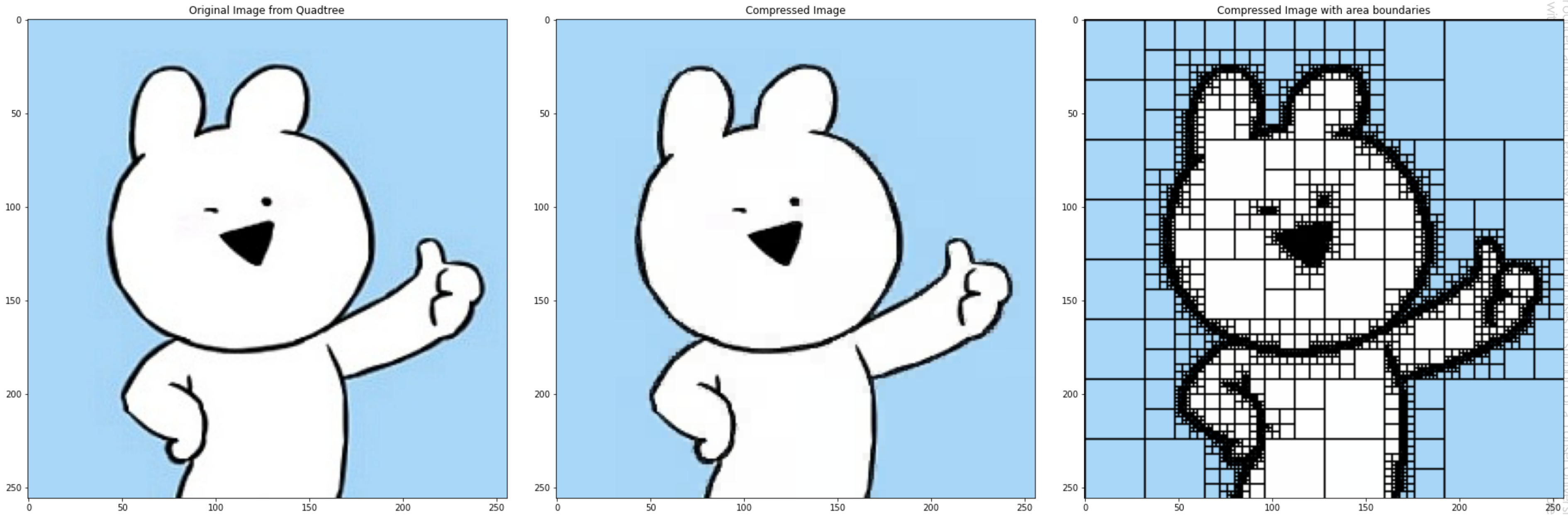
Compressed Image with threshold: 0.04

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.

Compressed Image with threshold: 0.00

Compressed Image with threshold: 0.00

This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other purposes and cannot be shared with others. Individuals who violate these terms and restriction are legally responsible for any violation of intellectual property laws.



This lecture material has been produced and published expressly for lecture purposes in support of a Yonsei University course. This material cannot be used for any other
purposes and cannot be shared wit
for any violation of intellectual property laws.