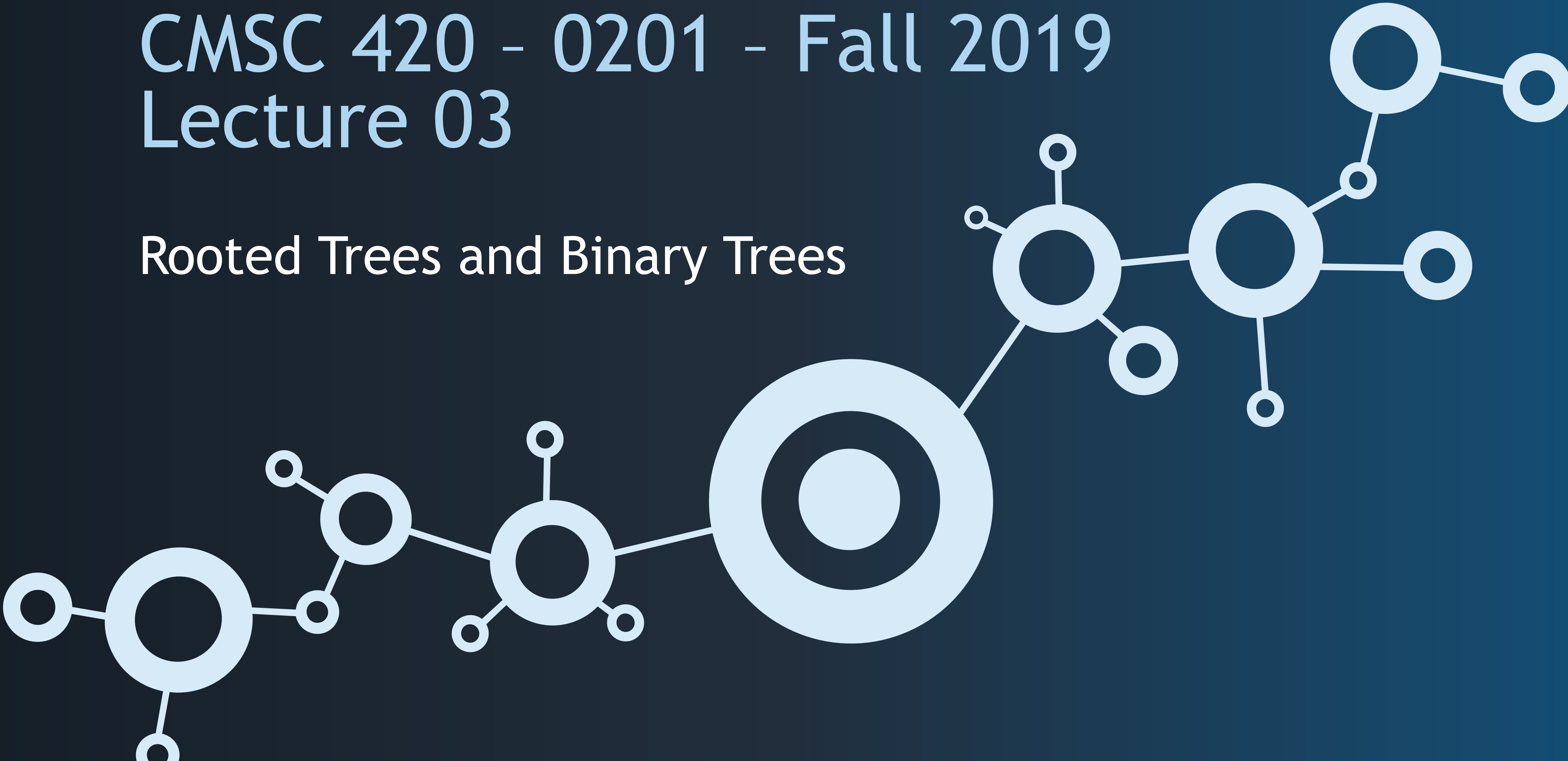
CMSC 420 – 0201 – Fall 2019
Lecture 03

Rooted Trees and Binary Trees



Tree Definition and Notation

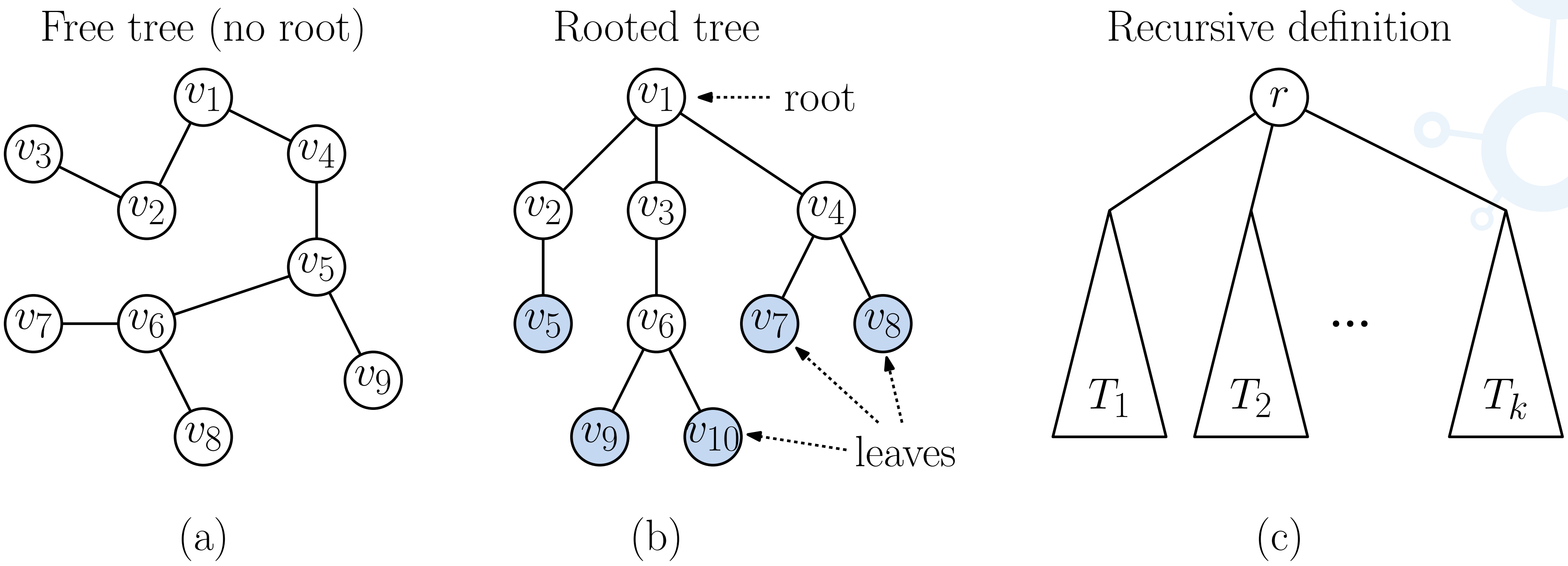
Graphs and Free Trees

- Graph: A graph $G = (V, E)$ is a finite set of nodes V and a set of edges E. Each edge is a pair of nodes
  - Directed graph: edge pairs are ordered
  - Undirected graph: edge pairs are unordered

Tree Definition and Notation

Graphs and Free Trees

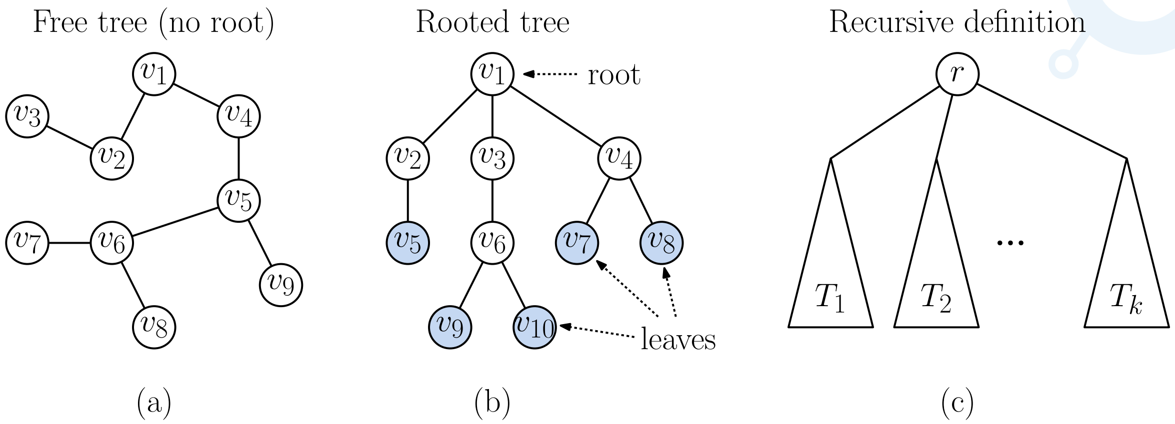
- Free Tree: A connected, undirected, acyclic graph
  - Example: Minimum spanning tree of an undirected graph



Tree Definition and Notation

Graphs and Free Trees

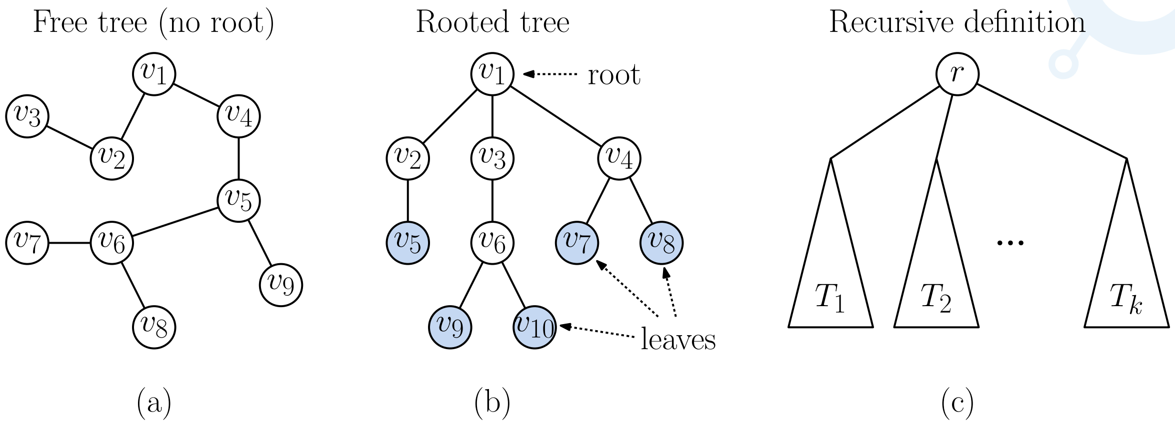
- Rooted Tree:
  - Designate a special node, called the root
  - As in a family tree, all other nodes are descendants of the root
  - Nodes with no descendants are called leaves
  - Nodes with one or more descendants are called internal nodes



Tree Definition and Notation

Recursive Definition

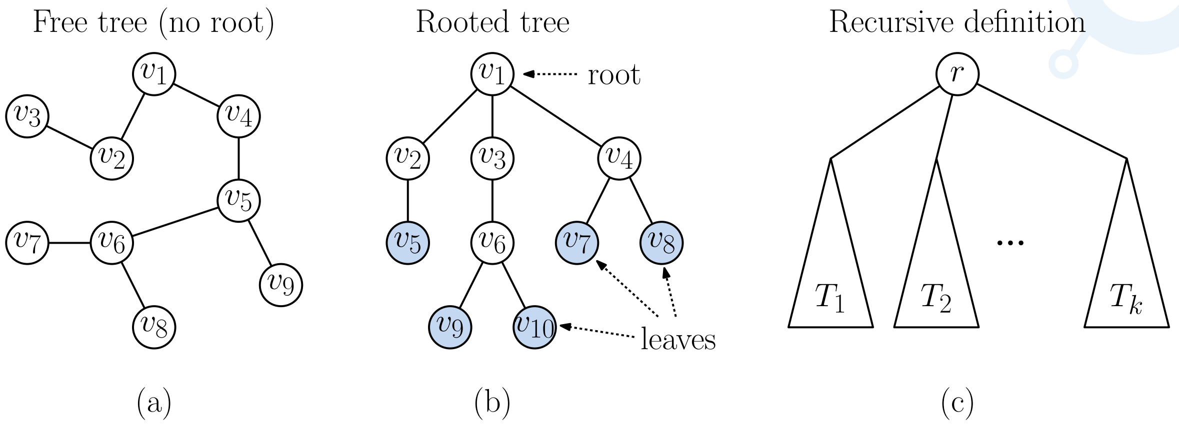
- Rooted Tree: (Recursive definition)
  - A single node is a rooted tree
  - Given rooted trees $T1, \ldots, Tk$, joining these trees under a common root node is a rooted tree
  - Note that this definition does not allow for empty trees
- Convention: When we say "tree", we mean "rooted tree" (not "free tree")



Tree Definition and Notation

Recursive Definition

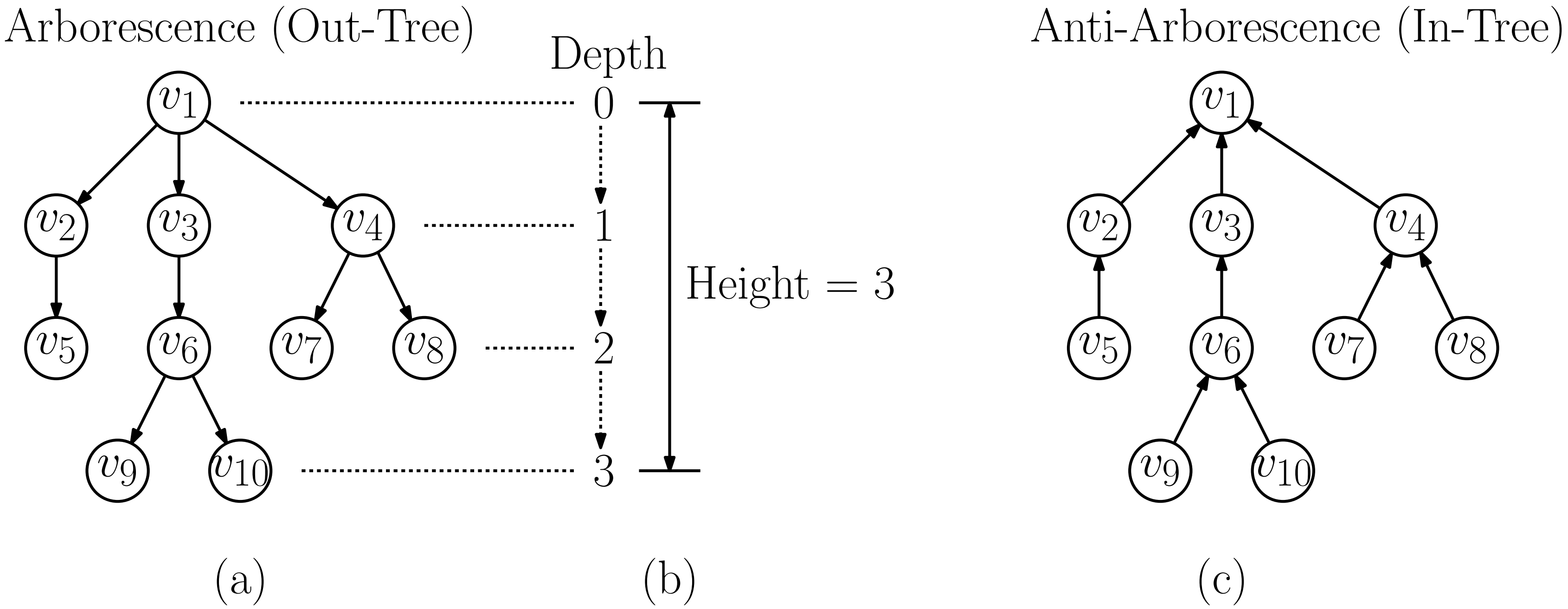
- Terminology:
  - From family trees: parent, child, sibling (all have the expected meaning)
  - Degree (of a node): is its number of children
  - Degree (of a tree): is the maximum degree of any node
  - Depth (of a node): is the length of path from root (root depth = 0)
  - Height (of a tree): is the maximum depth of any node
  - Ordered tree: Children are ordered (left to right)



Tree Definition and Notation

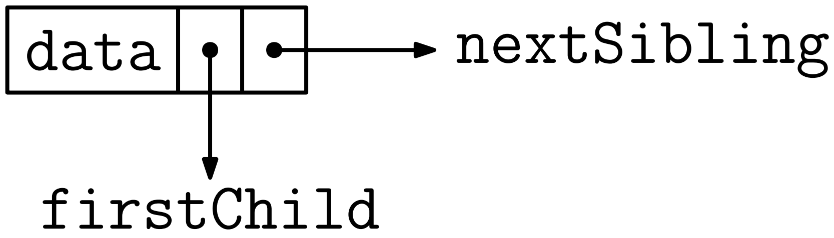
Arborescences – Out-trees and In-trees

- It is often handy to assign directions to the edges
- **Arborescence** (or **Out-Tree**): Edges emanate outwards from root
- **Anti-arborescence** (or **In-Tree**): Edges are directed inwards to the root

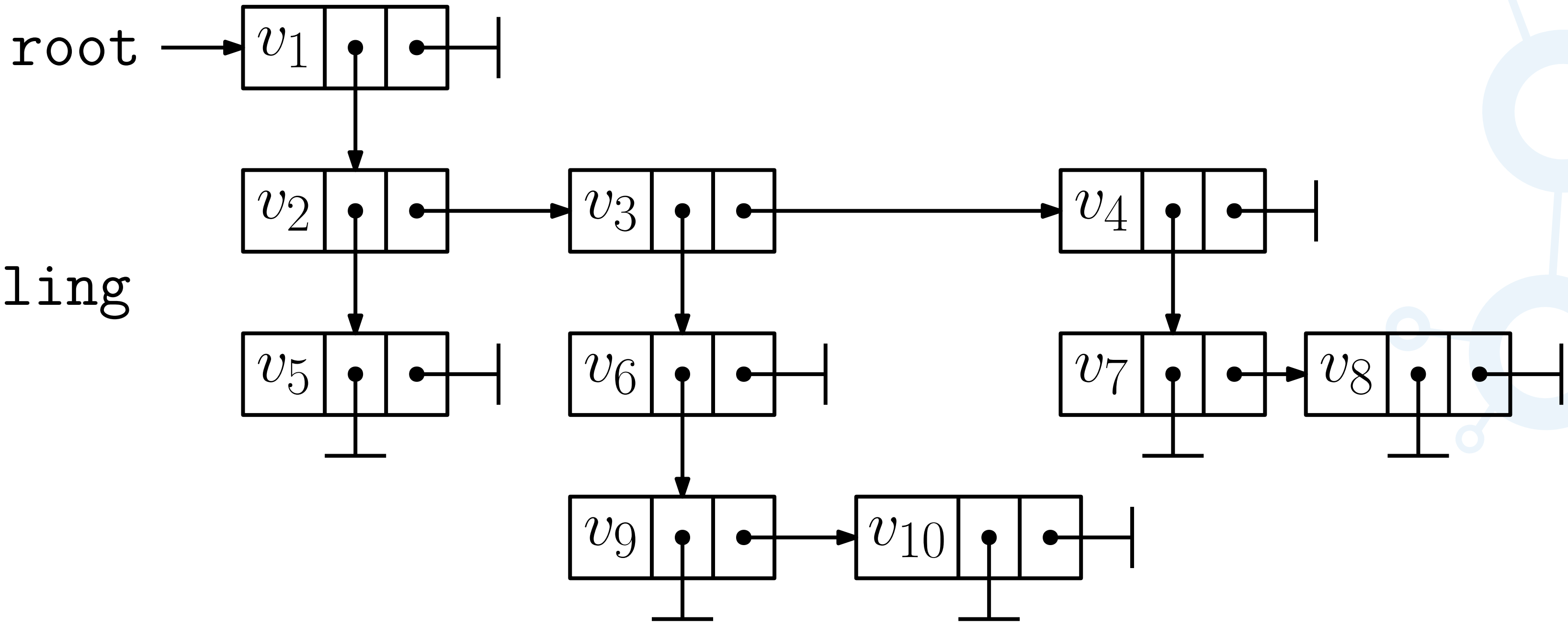


How to Represent Rooted Trees

Node structure (Binary-like)



(a)

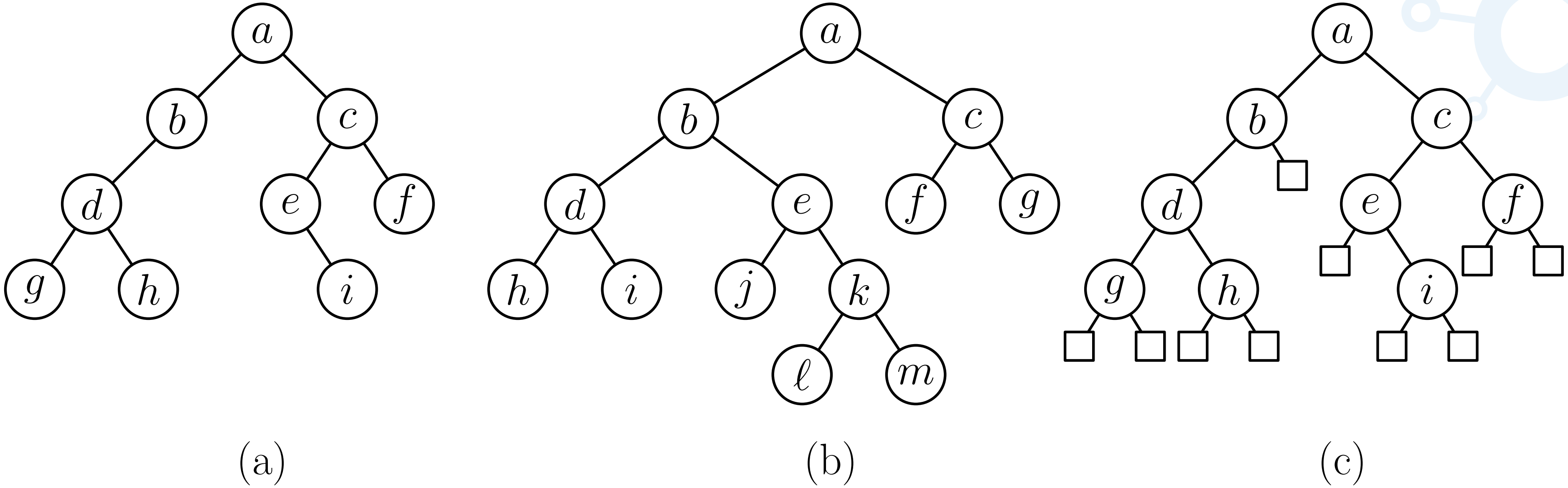


(b)

Binary Tree

Standard Definition

- Binary tree: A (possibly empty) rooted, ordered tree, where each internal node has two children, left and right
- Full binary tree: Every non-leaf node has exactly two children
- Extended binary tree: Replace empty subtrees special external nodes



(a) (b) (c)

Binary Tree

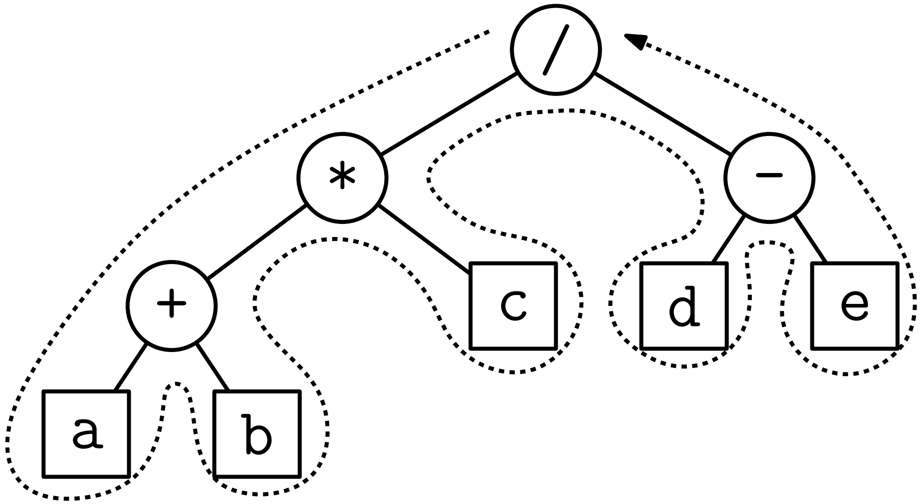
Java representation

```
class BinaryTreeNode<E> {
   private E                  entry;      // this node's data
   private BinaryTreeNode<E>  left;       // left child reference
   private BinaryTreeNode<E>  right;      // right child reference
   // ... remaining details omitted
}
```

- The entry type E can be filled in according to the application
- This is a minimalist representation. We might other information, such as a parent link
- Disclaimer: Java code in lectures is designed to be illustrative (and brief). It may be poorly structured and may contain errors. (If you find any, let me know.)

Tree Traversals

- Given tree with root r and subtrees $T_1, \ldots, T_k$:
- **Preorder**: Visit r, then recursively do a preorder traversal of $T_1, \ldots, T_k$
- **Postorder**: Recursively do a postorder traversal of $T_1, \ldots, T_k$ and then visit r
- **Inorder**: (for binary trees) Do an inorder traversal of T_L, visit r, do an inorder traversal of T_R.



Preorder: / * + a b c - d e

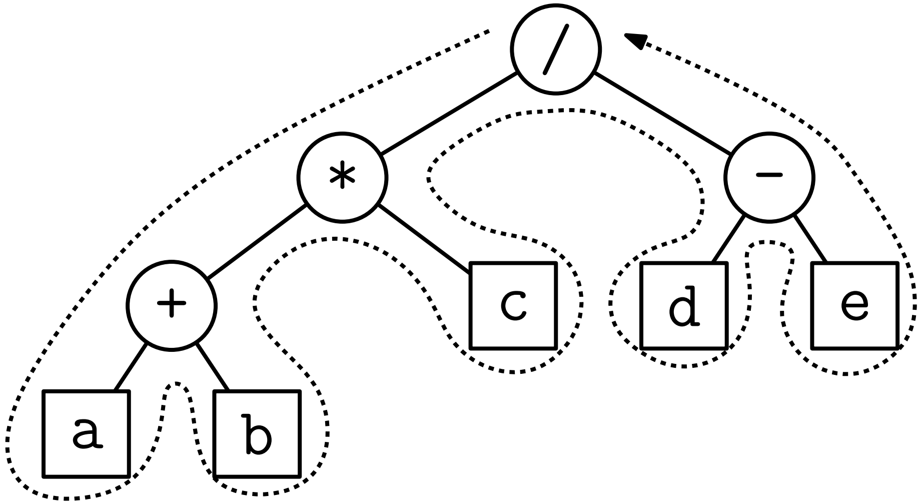
Postorder: a b + c * d e - /

Inorder: a + b * c / d - e

Tree Traversals

Java implementation of inorder traversal

```
void preorder(BinaryTreeNode v) {
       if (v == null) return;      // empty subtree - do nothing
       visit(v);                   // visit (depends on the application)
       preorder(v.left);           // recursively visit left subtree
       preorder(v.right);          // recursively visit right subtree
}
```



Preorder: / ∗ + a b c − d e

Postorder: a b + c ∗ d e − /

Inorder: a + b ∗ c / d − e

Extended Binary Trees

How many external nodes?

- **Theorem**: An extended binary tree with n internal nodes has $n + 1$ external nodes
- **Proof**: (By induction on n) Let $x(n)$ be the number of external nodes
  - $n = 0$: No internal nodes and 1 internal node. $x(0) = 1$, as desired
  - $n \geq 1$: Tree has a root and two subtrees, T_L and T_R. Let n_L and n_R be the numbers of internal nodes in each. We have $n = 1 + n_L + n_R$.
  - By induction, $x(n_L) = n_L + 1$ and $x(n_R) = n_R + 1$
  - Total number of external nodes is:

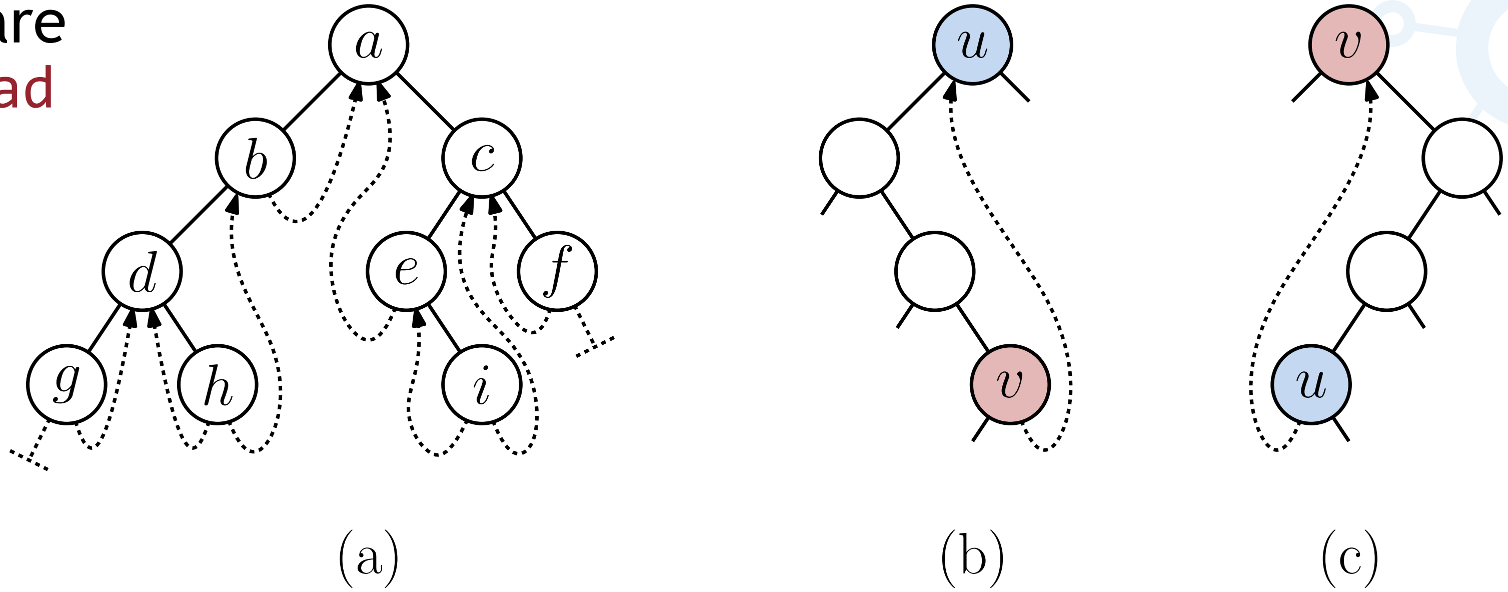
$$x(n_L) + x(n_R) = (n_L+1) + (n_R + 1) = (1 + n_L + n_R) + 1 = n + 1$$

- **Corollary**: It has a total of $2n + 1$ nodes

Threaded Binary Trees

Standard Definition

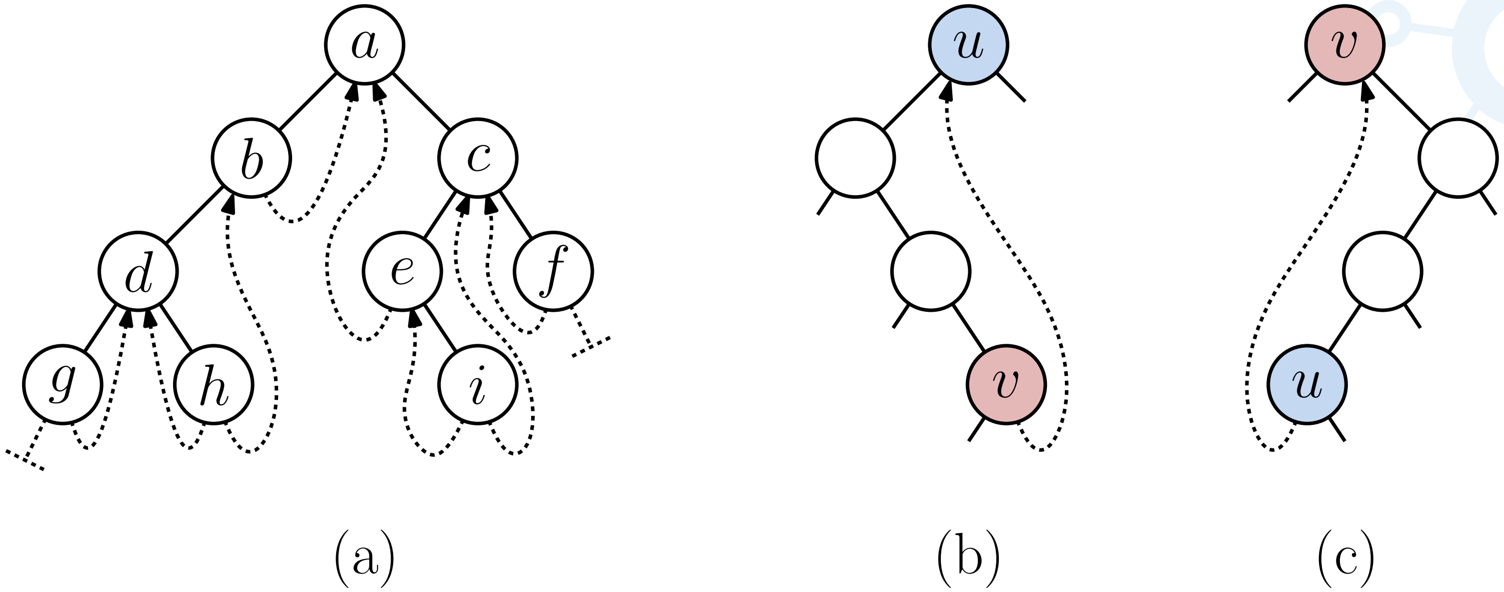
- Can we make better use of the null references? Help perform traversals!
- Left (null) child: Points to inorder predecessor
- Right (null) child: Points to inorder successor
- Add a mark bit so we know which links are real and which are threads (`u.left.isThread` and `u.right.isThread`)



Threaded Binary Trees

Standard Definition

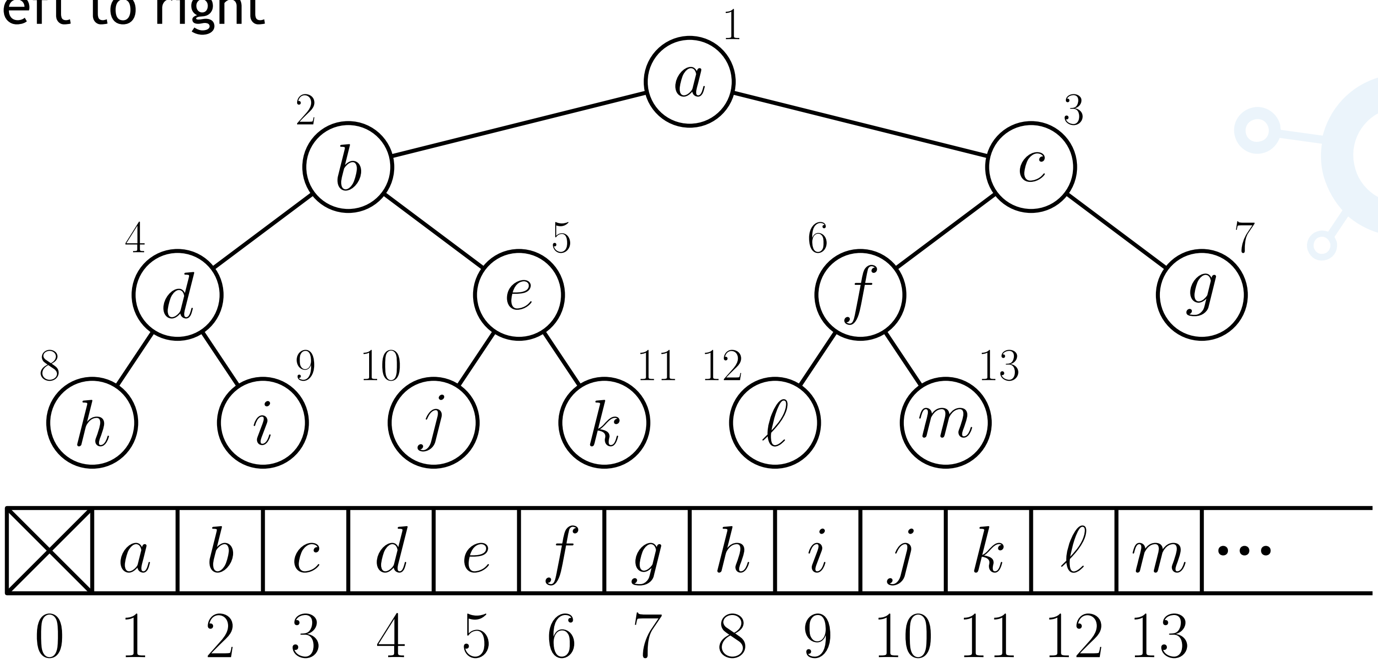
```
BinaryTreeNode inorderSuccessor(BinaryTreeNode v) {
      BinaryTreeNode u = v.right;              // go to right child
       if (v.right.isThread) return u;         // if thread, then done
       while (!u.left.isThread) {              // else u is right child
           u = u.left;                         // go to left child
       }                                       // ...until hitting thread
       return u;
   }
```



Complete Binary Trees

…and array allocation

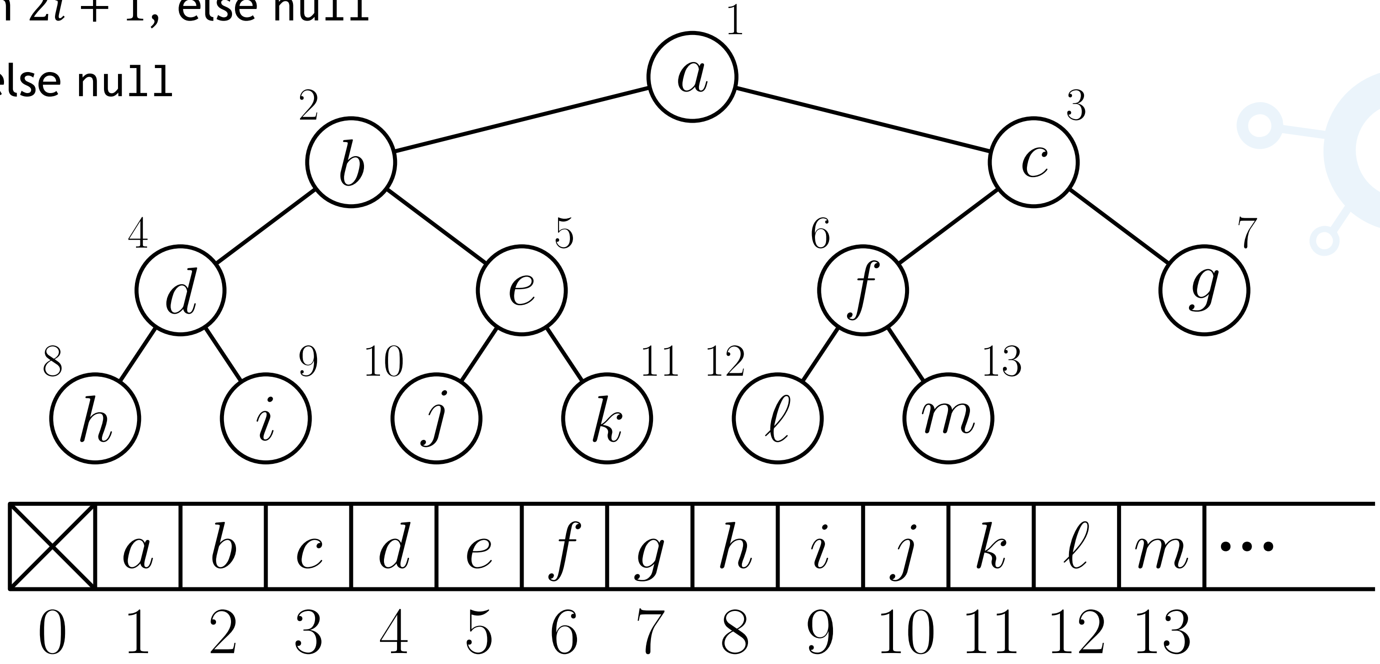
- Can we allocate binary trees in an array, without pointers?
- Yes, but the tree needs to be really full
- Complete Binary Tree: Every level of the tree is completely filled, except possibly the bottom level, which is filled from left to right



Complete Binary Trees

…and array allocation

- We can allocate the nodes of a complete binary tree in an array as follows:
  - Number the nodes level by level from 1 to n, and store in array $A[1 \ldots n]$
  - leftChild(i): if $(2i \le n)$ return $2i$, else `null`
  - rightChild(i): if $(2i+1 \le n)$ return $2i+1$, else `null`
  - parent(i): if $(i \ge 2)$ return $\lfloor i/2 \rfloor$, else `null`



Summary

- Rooted trees – Definition, terminology and representation
- Binary trees - Definition and terminology
- Node representation
- Tree traversals
- Extended binary trees (and number of external nodes)
- Threaded binary trees
- Complete binary trees and array allocation