

```
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
```

```
#define MAXPAROLA 30
#define MAXRIGA 80
```

```
int main(int argc, char *argv[])
```

```
{
    int freq[MAXPAROLA]; /* vettore di contatori
    delle frequenze delle lunghezze delle parole */
    char riga[MAXRIGA];
    int i, inizio, lunghezza;
    FILE *f;
```

```
for(i=0; i<MAXPAROLA; i++)
    freq[i]=0;
```

```
if(argc != 2)
```

```
{
    fprintf(stderr, "ERRORE, serve un parametro con il nome del file\n");
    exit(1);
}
```

```
f = fopen(argv[1], "r");
if(f==NULL)
```

```
{
    fprintf(stderr, "ERRORE, impossibile aprire il file %s\n", argv[1]);
    exit(1);
}
```

```
while( fgets( riga, MAXRIGA, f ) != NULL )
```



# Trees

## Definitions

Stefano Quer

Dipartimento di Automatica e Informatica

Politecnico di Torino

# Rooted trees

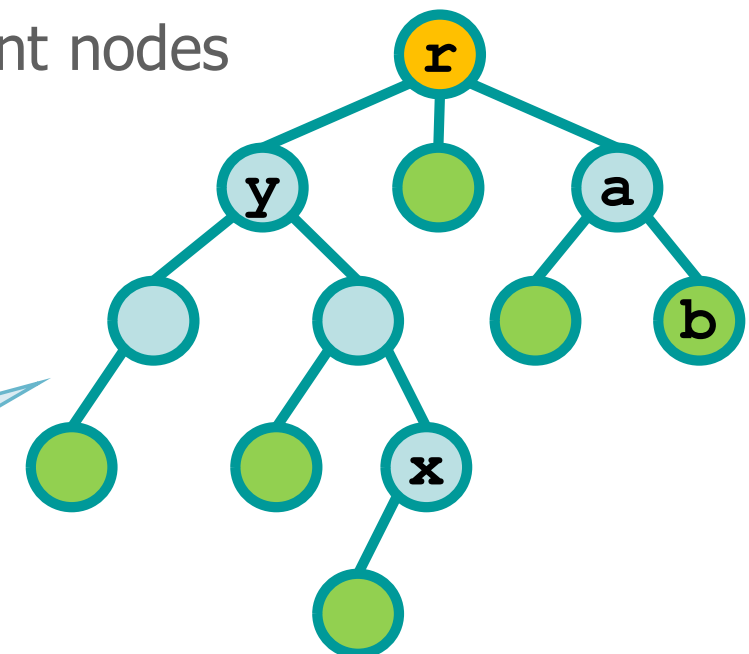
❖ A rooted tree is a tree where there is a node  $r$  called root

➤ Parent/child relationship

- $y$  is an ancestor of  $x$  if  $y$  belongs to the path from  $r$  to  $x$ . In this case  $x$  is a descendant of  $y$
- $y$  is a proper ancestor of  $x$  iff  $x \neq y$
- Parent and a child are adjacent nodes

➤ The root has no parent

➤ Leaves have no children

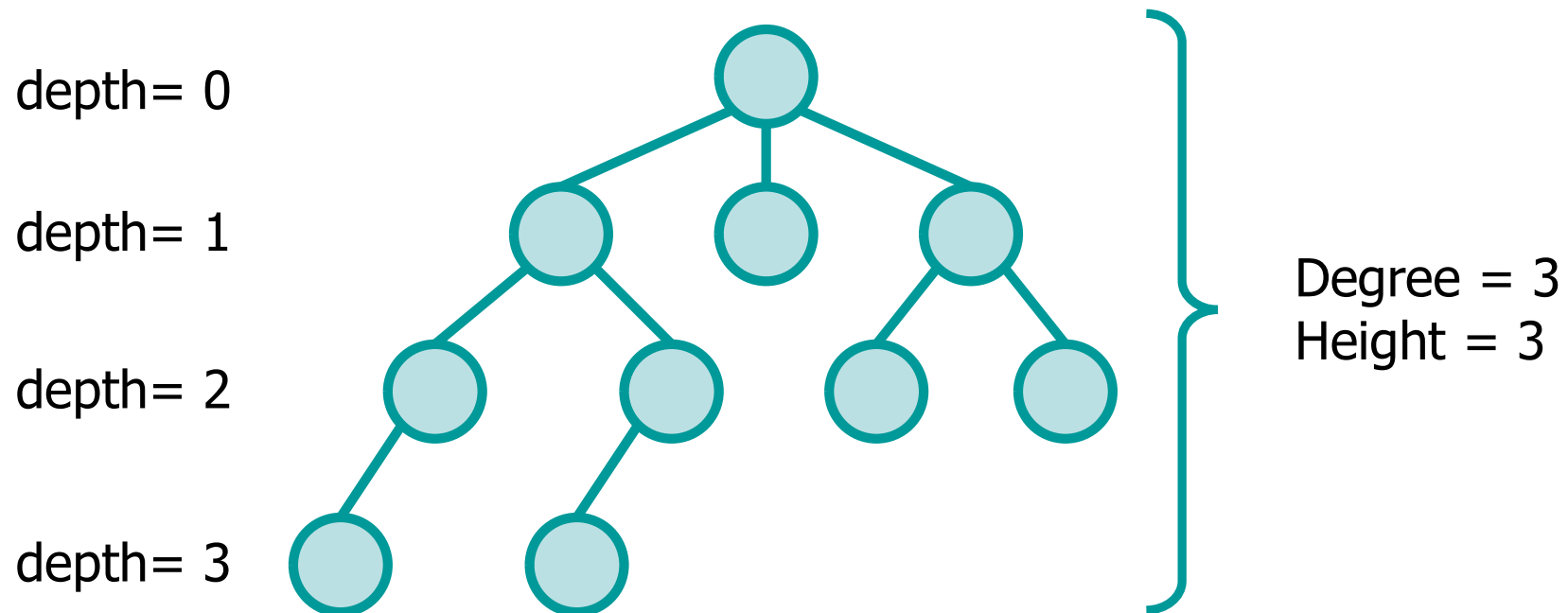


$y$  ancestor of  $x$   
 $x$  descendant of  $y$   
 $a$  parent of  $b$   
 $b$  child of  $a$

## Properties of a rooted tree

❖ Given a rooted tree  $T$  the following are common definitions

- Degree ( $T$ ) = maximum number of children
- Depth ( $x$ ) = length of the path from the root to  $x$
- Height ( $T$ ) = maximum depth of a node

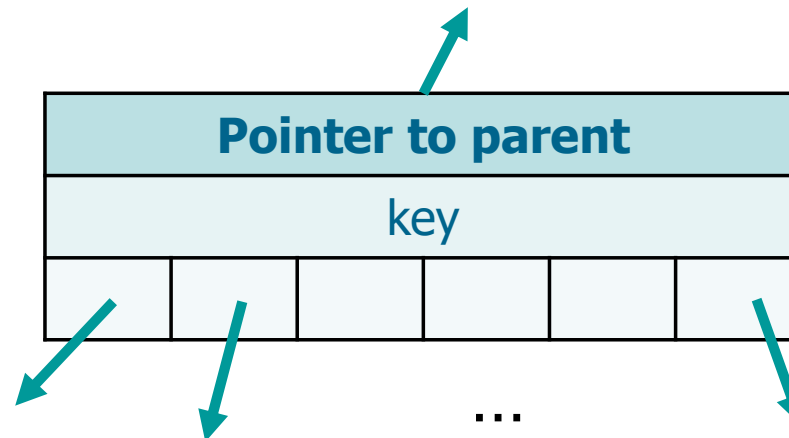


## Representation of a tree

- ❖ There are at least two representations for nodes of a tree of degree  $k$ 
  - Each node may store a pointer to the parent, the key, and  $k$  pointers to  $k$  children

The pointers to the father is optional

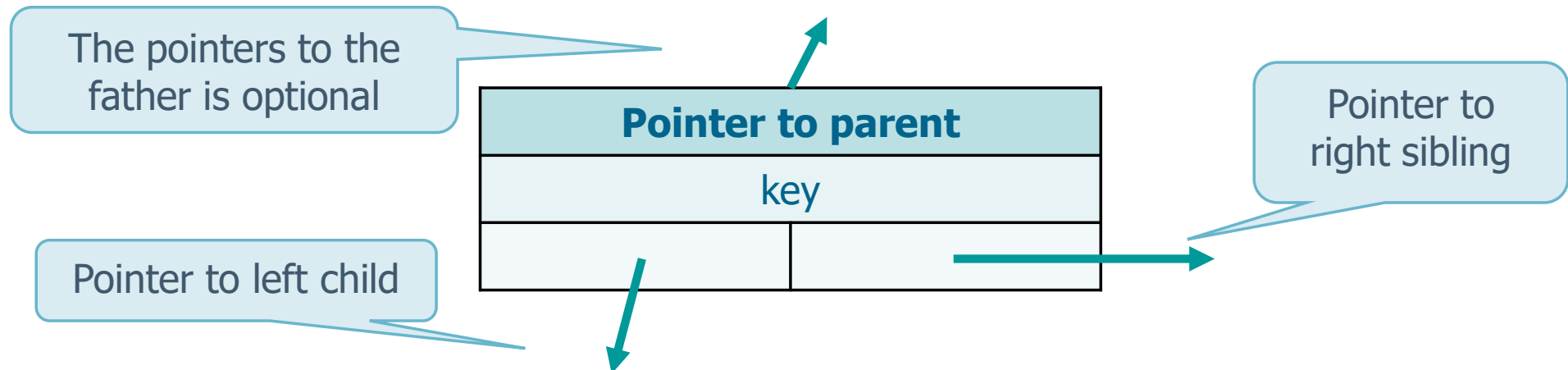
Pointers to  $k$  children.  
Possibly NULL



- Unefficient if only few nodes have indeed degree  $k$ 
  - Space is allocated for all  $k$  pointers, but many are NULL)

## Representation of a tree

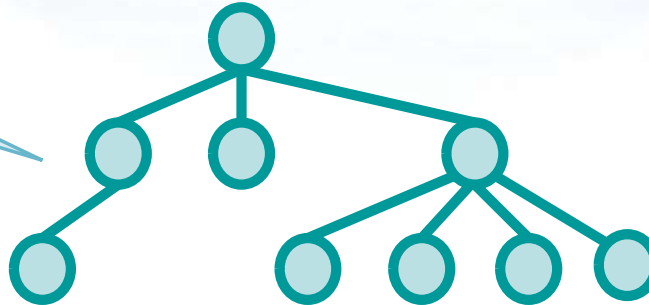
- Each node may also store a pointer to parent, the key, 1 pointer to left child, 1 pointer to right sibling



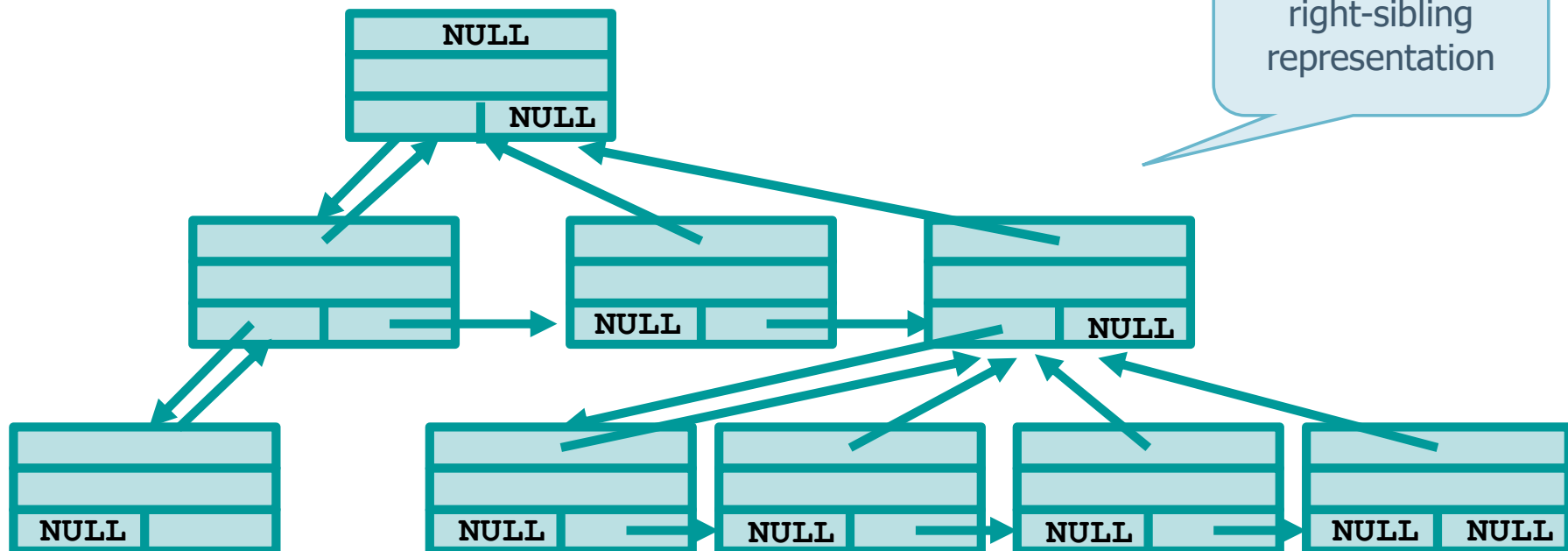
- Efficient, as each node specifies always 2 pointers, no matter the degree of the tree

# Representation of a tree

Standard  
representation



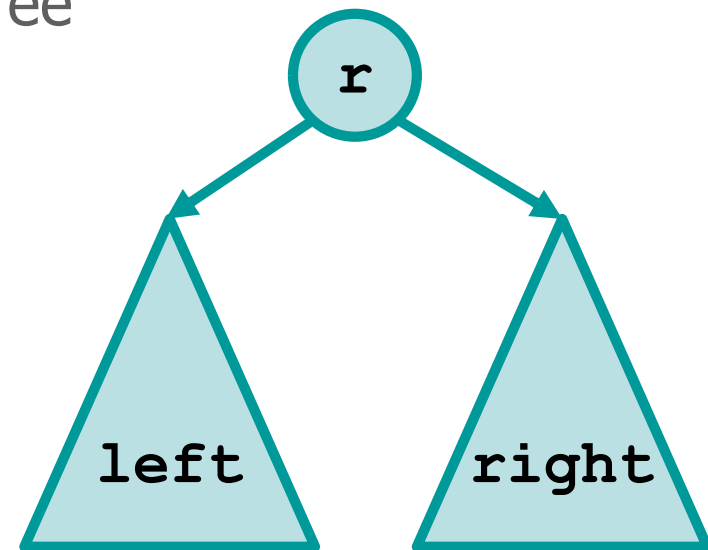
Left-child  
right-sibling  
representation



# Binary trees

## ❖ Definition

- Tree of degree 2
- Recursively T is
  - Empty set of nodes
  - Root, left subtree, right subtree

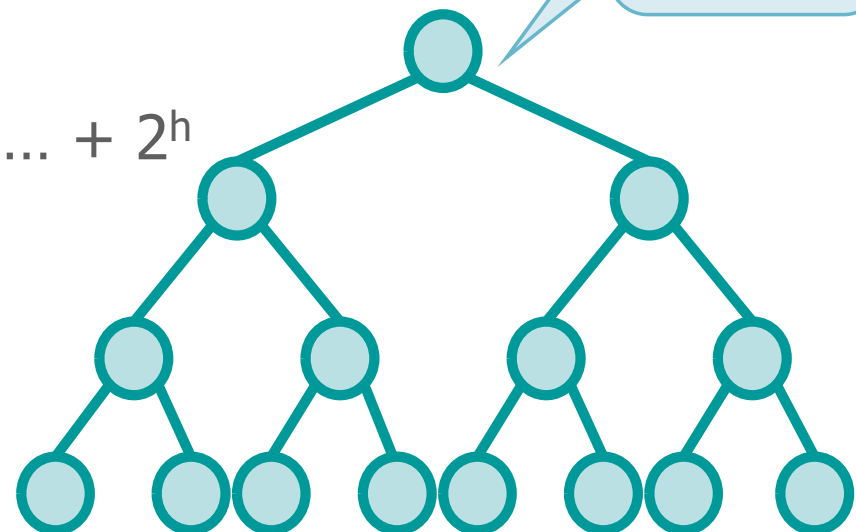


# Complete Binary Trees

- ❖ A complete binary tree must satisfy two conditions
  - All leaves have the same depth
  - Every node is either a leaf or it has 2 children
- ❖ In a complete binary tree of height  $h$ 
  - The number of leaves is  $2^h$
  - The number of nodes is

$$\begin{aligned}\blacksquare \sum_{0 \leq i \leq h} 2^i &= 2^0 + 2^1 + 2^2 \dots + 2^h \\ &= 2^{h+1} - 1\end{aligned}$$

$h = 3$   
8 leaves  
15 nodes

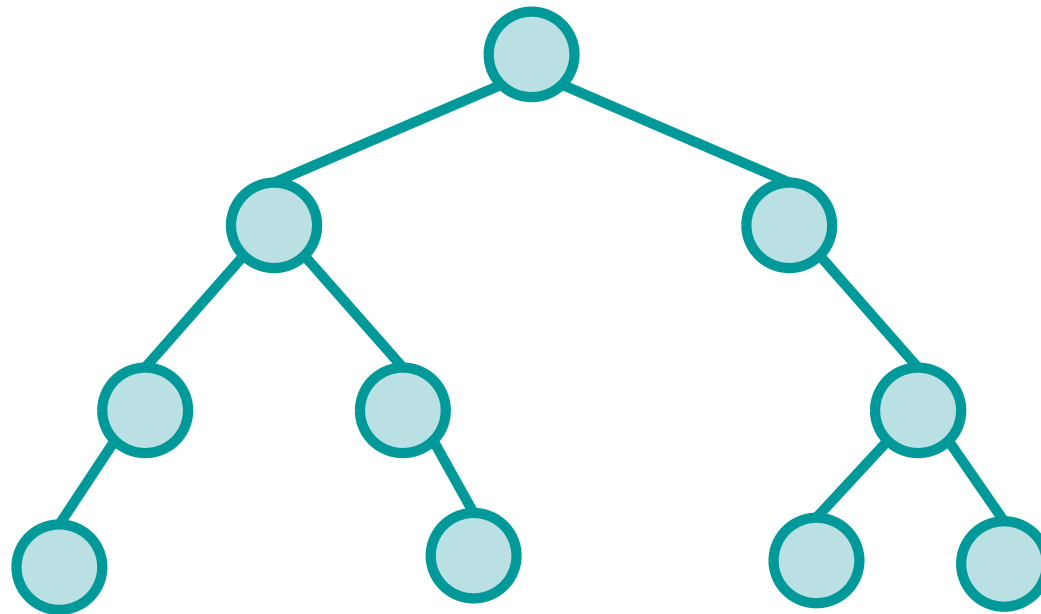


Finite geometric  
progression with ratio = 2



## Balanced binary trees

- ❖ In a balanced binary tree all paths root-leaves have the same length



- If  $T$  is complete, then  $T$  is also balanced
- The opposite is not necessarily true

## Balanced binary trees

- ❖ A binary tree is said to be almost balanced if the length of all paths from root to leaves differs at most by 1

