Quad Trees

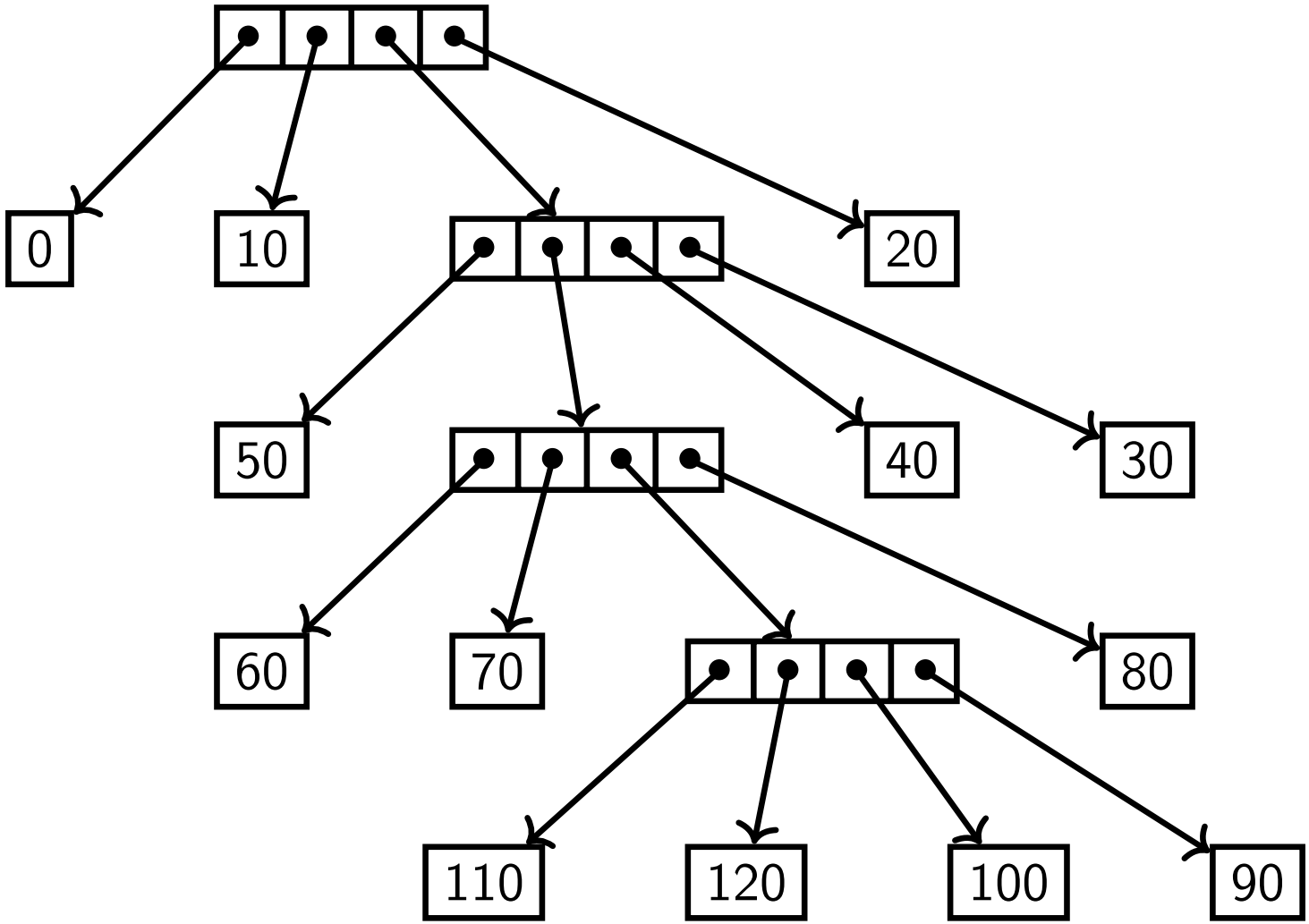
Quad Trees

A *Quad Tree* is a particular kind of tree that differs from the binary tree that we have looked at so far in two respects:

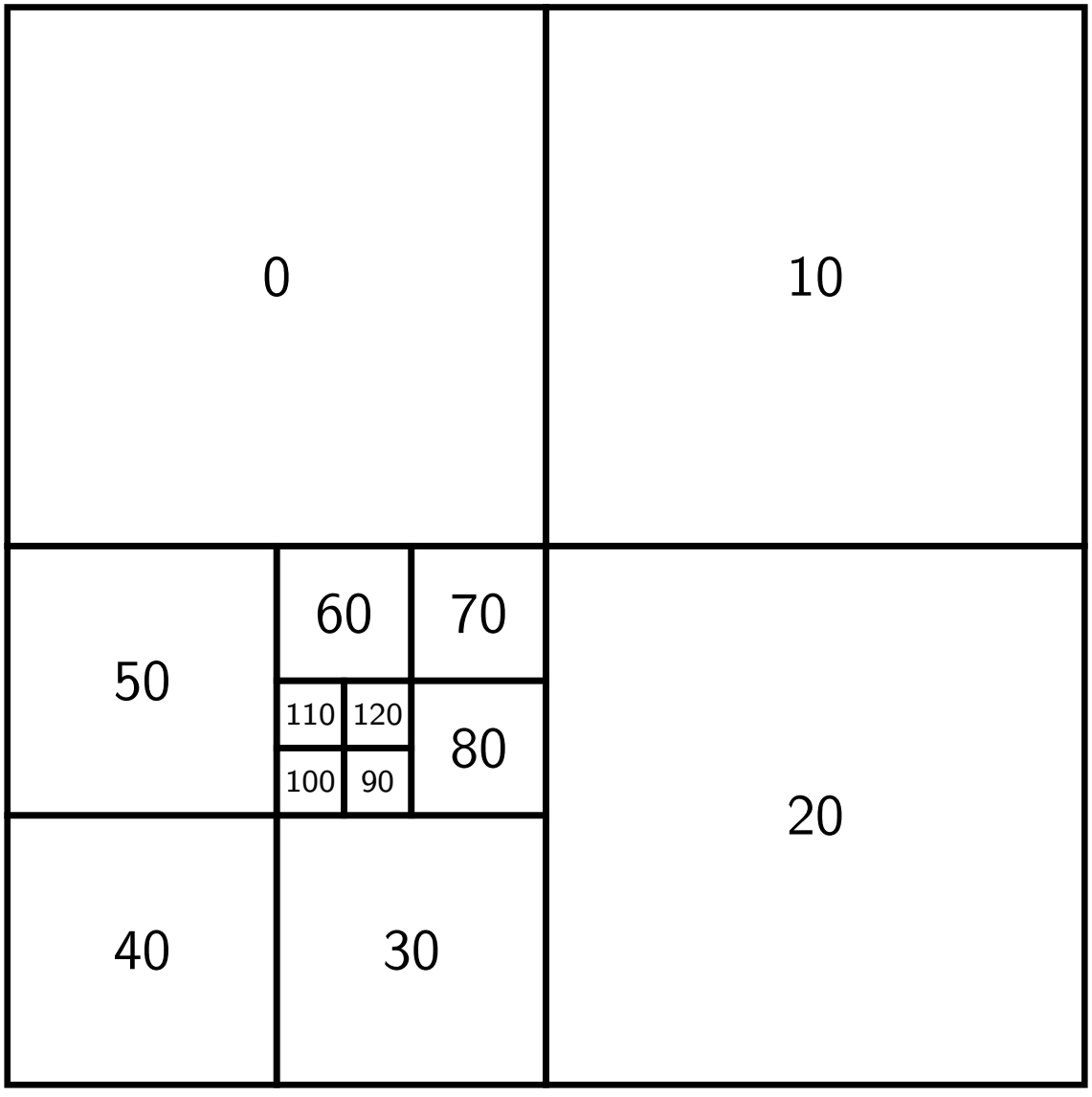
- The values are ONLY at the leaf level: there are no values in internal nodes of the tree
- Internal nodes have 4 children

This data structure is particularly useful in representing and manipulating some kinds of 2-dimensional data such as images. A typical application is in image compression.

Quad Trees



Quad Trees



Quad Tree ADT

- Constructors:
  - `baseQT` : returns a single, leaf node quad tree with a value
  - `MakeQT(luqt, ruqt, llqt, rlqt)` : returns a new quad tree built from four sub-quad trees.
- Accessors:
  - `isValue(qt)` : return true if `qt` is a value node quad tree, otherwise returns false
  - `lu(qt)` : returns the left upper sub-quad tree of `qt` [1]
  - `ru(qt)` : returns the right upper sub-quad tree of `qt` [1]
  - `ll(qt)` : returns the left lower sub-quad tree of `qt` [1]
  - `rl(qt)` : returns the right lower sub-quad tree of `qt` [1]

If `qt` is a value node quad tree, then we conventionally refer to the value stored in the node as `qt`, rather than define another accessor `value(qt)` for that purpose.

[1] Triggers an error if the `qt` is a value node quad tree

Example

```
1 rotate(qt) {
2   if ( isValue(qt) )
3     return qt
4   else
5     return makeQT( rotate(rl(qt)),
6                    rotate(ll(qt)),
7                    rotate(ru(qt)),
8                    rotate(lu(qt)) )
```