Tries

Johan Montelius

KTH

HT23

Binary trees

Binary trees

- sorted tree, find any key in $O(lg(n))$

Binary trees

- sorted tree, find any key in $O(lg(n))$
- heap, access highest priority in $O(1)$, add and remove in $O(lg(n))$

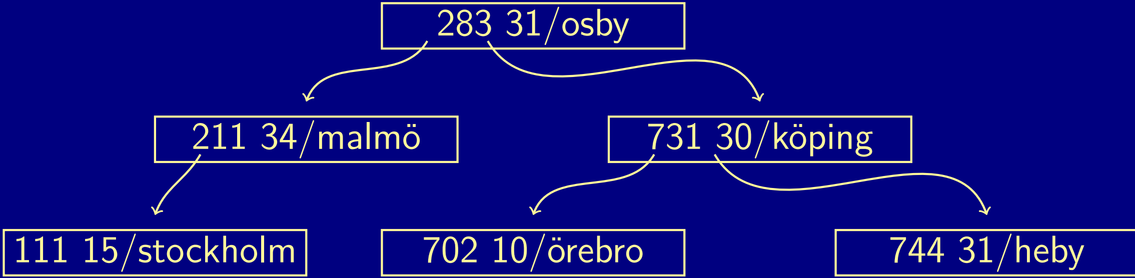
Binary trees

- sorted tree, find any key in $O(lg(n))$
- heap, access highest priority in $O(1)$, add and remove in $O(lg(n))$
- more advanced version to keep balanced

Binary trees

- sorted tree, find any key in $O(lg(n))$
- heap, access highest priority in $O(1)$, add and remove in $O(lg(n))$
- more advanced version to keep balanced
- could be implemented using an array

typical usage



k-arry trees

k-arry trees

A tree with more than two branches in a node.

k-arry trees

A tree with more than two branches in a node.

more of the same

k-arry trees

A tree with more than two branches in a node.

more of the same

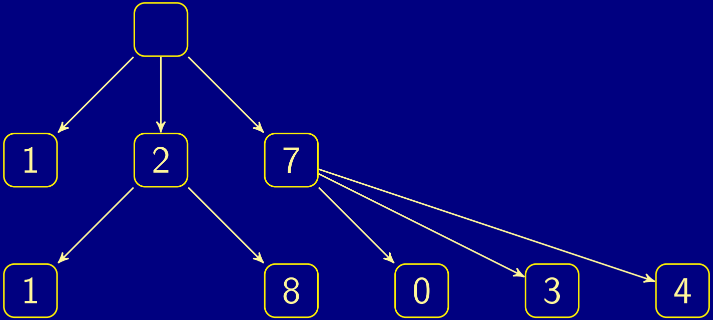
keys and values are present in each node of the tree

what if...

what if...

The keys are implicit in the path from the root to the node of the value.

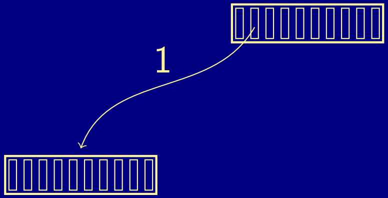
the path is the key



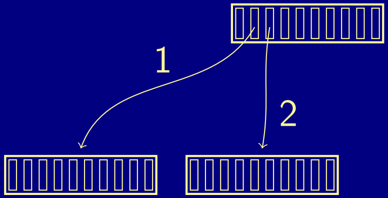
let's do like this

$\boxed{\text{□□□□□□□□□□}}$

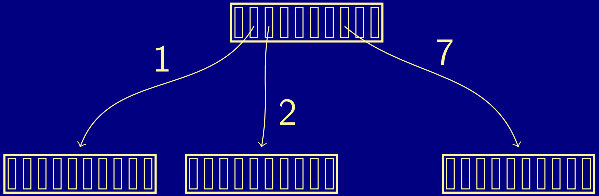
let’s do like this



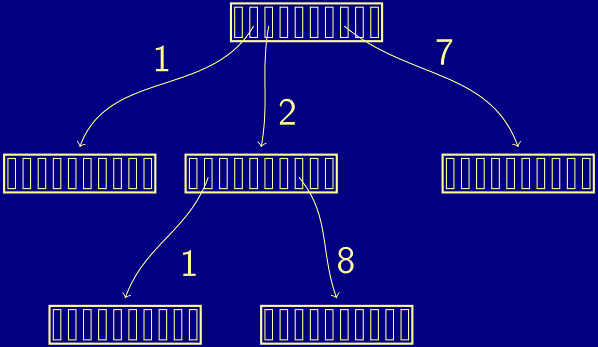
let's do like this



let's do like this



let's do like this



usage

usage

- we need to be able to divide the "key" into sub-keys

usage

- we need to be able to divide the "key" into sub-keys
- dictionary, each level is a character, a path is a word

usage

- we need to be able to divide the "key" into sub-keys
- dictionary, each level is a character, a path is a word
- words will share paths: small, small-e-r, small-e-s-t

hashed array mapped trie

- Use a hash function to produce a fixed size key.

hashed array mapped trie

- Use a hash function to produce a fixed size key.
- Divide key into fixed sized sub-keys (5-bits).

hashed array mapped trie

- Use a hash function to produce a fixed size key.
- Divide key into fixed sized sub-keys (5-bits).
- Each node consist of a bit pattern (32-bits) and an array of node pointers.

hashed array mapped trie

- Use a hash function to produce a fixed size key.
- Divide key into fixed sized sub-keys (5-bits).
- Each node consist of a bit pattern (32-bits) and an array of node pointers.

hashed array mapped trie

- Use a hash function to produce a fixed size key.
- Divide key into fixed sized sub-keys (5-bits).
- Each node consist of a bit pattern (32-bits) and an array of node pointers.

The original key is not given by the path.