6. March 2008

Digital search [Radix search]:

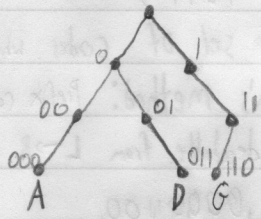
Simplest case: Each key has fixed length (ℓ) in bits

Ex. Key	Representation
A	000
B	001
C	010
D	011
E	100
F	101
G	110
H	111

(Binary)

Naïve Trie [Uncompacted trie] representing {A, D, G}

- $\wedge$ (4 elems) 0 ⇒ null trie, otherwise a trie whose left subtree



- Can be viewed as DFA
- Start @ root, left if 0, right if 1
  - ↳ accept iff end @ node.
- Searching:

```
member(Trie t, Key k)   // K∈t? K:=k0k1...kℓ-1
  return m_helper(t,k,0);

m_helper(Trie t, Key k, int i){ //i is depth in trie
  if (i==ℓ) return true; //ℓ==depth ∵ using fixed length.
                         // if we got to bottom, found key.
  if (t==NULL) return false;
  if (ki==0) return m_helper(t.left, k, i+1);
  else       return m_helper(t.right, k, i+1);
}
```

- Insertion: Traverse trie, take path defined by key building [1, ℓ] nodes along the way by filling in path as necessary.

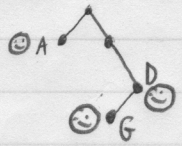
```
insert(Trie t, Key k) return i_helper(t,k,0);

i_helper(Trie t, Key k, int i) {
  if (i==ℓ) return t;
  if (t==NULL) r=new Trie(NULL,NULL);
  else         r=t;
  if (ki==0) return i_helper(r.left, k, i+1);
  else       return i_helper(r.right, k, i+1);
```

- Deletion: Delete node corresponding to key, then delete parents that don't have children. (when recursing up delete yourself if you have no children)

Variants on the Naïve Binary Trie:

① Variable-length codes for keys

- A=0, B=1, C=10, etc.



- Need to mark nodes corresp. to members (use ☺)
  - ↳ Node representation: | left | right | ☺/☹ |

- Searching:

```
m_helper(t,k,i){
  if (t==NULL) return false;
  if (i==|k|) return t.☺;
  if (ki==0) return m_helper(t.left,k,i+1);
  else       return m_helper(t.right,k,i+1);
i_helper(t,k,i){
  if (t==NULL) r=new(NULL,NULL,☹);
  else         r=t;
  if (i==|k|) t.☺=true;
  if (ki==0) ret i_helper(t.left,k,i+1)
  else                    right
```

Prefix codes [Prefix-free codes]

- String coding for keys $\ni$ no key a prefix for another.
- Ex.

A	01
B	001
C	000
D	100
E	1010
F	1011

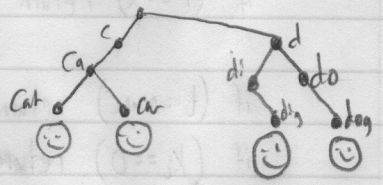
Not an intuitive representation. Most codes not prefix codes.

- Same as set of codes where ☺ only @ leaves, never interior.
- Huffman's method: Prefix code w minimal expected search time in a Naive Binary Trie
- Uniquely decodable from L→R
  - ex. 0000100 → C A D

Data compression: Find sequence of bits that encodes sequence of keys
Minimize E(length) per key w/ assumption that keys have fixed, indep. probabilities

Non-binary Tries:

- Keys are strings from a finite alphabet.
- Ex. Σ = a-z, {cat, car, dog, dig}

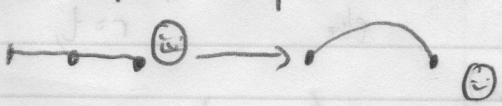


- Instead of just 2 children, have $|\Sigma|$ children

Other variants:

Slightly compacted tries = $Trie_{N+}$
More compacted tries = Patricia tries
Patricia tries can also refer to something else (a complex representation)

Slightly compacted: Compress a chain of unhappy nodes leading to a leaf.



Patricia compaction: All unhappy chains compressed. (replaced by 1 edge)
Only unhappy nodes are nodes leading to >1 happy.