Project Report – Longest Prefix Match

10/20/09
Deepak Konidena
Mandeep Singh

The project is about constructing two types of tries as primary data structures for holding routing information.

a) Binary Trie
b) Multibit Trie (with a fixed stride of 2)

Binary Trie contains the following data structures. BiTrie and TrieNode. BiTrie stores the root node of the trie and a current node pointer curRoot, which helps in navigating along the trie during insertion.BiTrie also has a size element which keeps track of the number of nodes present at a given instant.
TrieNode is the node data structure which contains two child pointer nodes left and right along with a data pointer that stores the nexthop value at that node.

Multibit trie contains the following data structures. mTrie and mTrieNode. mTrie stores the root node along with the size of the trie and current node pointer curRoot. curRoot performs the same function as curRoot of BiTrie.
mTrieNode contains four child pointers first, second, third and fourth and two data pointers. The two data pointers serve as primary nexthop and secondary nexthop. Primary nexthop pdata contains the normal nexthop value , whereas secondary nexthop sdata contains the pushed down nexthop value. When odd length prefixes are encountered, the nexthop is pushed down to its children.(00 and 01 for 0* and 11 and 10 for 1*) .

The following functions are implemented for operations on the Binary trie.

a) BiTrieInit
b) BiTrieInsert
c) updateNode
d) deleteNode
e) newnode

BiTrieInit initializes BiTrie data structure and sets the root pointer. BiTrieInsert inserts the nexthop values into individual nodes based on their prefixes. updateNode updates the node with the announced nexthop, whereas deleteNode either deletes the node if it is the leaf or sets its nexthop value to NULL if it is not a leaf node. newnode creates a node and returns its pointer to the caller.

The following functions are used for file reading and parsing operations.

a) getPrefix
b) ParseUpdate
c) ParseLookup

ParseLookup and ParseUpdate read the files lookup.txt and update.txt and parse them. While getPrefix returns the binary prefix of the string format of IP address.

getLongestPrefixMatch returns the nexthop value of the longest prefix match of the prefix.

Multibit trie has the following functions

a)mTrieInit
b)mTrieInsert
c)mnewnode

During multi-bit trie insertion, nexthop values of the prefixes that have odd lengths are pushed down to their children. It is copied as a secondary nexthop (sdata)in its children. Nexthop values of prefixes that have even lengths are copied as a primary nexthop value (pdata) in their nodes. And during longest prefix match, both the nexthop values of the node are searched for, returning whichever is present in the order primary being first.

Multi bit updates are done in a similar way. The odd length prefixes update the secondary nexthop values of their children, while the even length prefixes update the primary nexthop values of themselves.

In multibit deletes, if odd length prefix is to be deleted the secondary next hop values of the children are deleted , whereas if even length prefix is to be deleted the primary nexthop value of the respective node is deleted.

The following functions help calculate the mean, median, minimum and maximum time taken to update and perform longest prefix match.

a) getMedian
b) getMean
c) getMin
d) getMax

Assumptions:

1) The unibit and multibit tries are constructed , updated and deleted simultaneously. And hence the output is two messages printed in succession. Incase of multibit, updation and deletion can cause updating and deleting more than node, and that's because of the odd length prefixes and maintaining two copies of nextHop.
2) The total number of nodes count excludes the root node.

Usage:

```
./lpm <file for loading prefixes>
```