Workshop Leaplist Todo’s

1. **Talk to moshe.** Range queries – the interface in the code states that it only traverses the structure. It doesn’t return a range of requested values. We understand that for benchmarking purposes there’s not really a difference, since from that point on it’s one threaded code.
2. Node.Low id needed?
3. Node members : Low,High,Count change to long . level to byte. Check modifiers also (protected/public etc.)
4. Metadata in tire???
5. TrieMetadata . prefix\_length & prefix change to long that holds 4 lower bits as prefix\_length and prefix to upper 60. Add a property that gets prefix\_length and that gets prefix.
6. Remove big chunks of code from constructors to functions
7. trie\_create\_from\_array in loops why are the variables are volitale ? seems like only one thread will run on trie funcs so why volatile . also ln. 241 in trie.c , the if could be removed and set outside the loop. Notice : No volitaile in java implementation .
8. struct trie\_metadata\_t.

typedef struct {

unsigned int prefix\_length : 4;

unsigned long prefix : 60;

} trie\_metadata\_t;

this could take a lot more memory than intended, needs to be changed to unsigned long and unsigned long. Check if current configuration indeed takes 8 bytes and not 12/16. Check on nova.

1. New leaplist() doesn’t initialize trie.
2. Insert – overwrites means that insert always happens. If overwrite = 1 , if an update is required perform the update. if overwrite =0 , if an update is required don’t perform the update.

Explanation of how Trie works.

Building a trie from a given array :

The trie iterates through the tree by checking 2 hex digits at a time (Most significant).

With one exception , when there’s a total number of digits which is odd, the upmost layer will containing 16 nodes (as opposed to normally 256) to represent 16 hex possibilities of one digit.

Prefix is the Most significant joint hex digits . when it’s 14 it means we have only 2 hexes that’s why 256 nodes are create. 15 and we have only 1 hex that’s different, thus we create 16 nodes.