**A Brief summary of the SkipTrie, SkipList and the X-Fast-Trie performance**

Victor Jakubiuk

**In short:** the running time of the X-Fast-Trie is about 20% better than the running time of the SkipList (it’s due to the L2 cache misses). Unfortunately, it’s too little for the SkipTrie to improve meaningfully.

**Here’s how the conclusion has been reached:**

(Please see the spreadsheet with data & all graphs for other performance counters).

1. The X-Fast-Trie (with a regular, GCC/C++11 unordered map as a hash table) is about 20% faster than the SkipList (of the same size):

And this is mostly due to L2 cache misses (= L3 cache accesses):

2. Replacing top rows in the SkipList with a “small” X-Fast-Trie, barely improves the performance of the SkipTrie (since the trie is “small” and it’s anyway only 20% faster):

3. Increasing the size of the top X-Fast-Trie brings the SkipTrie performance characteristics (both the running time and the cache perf.) closer the performance of a full X-Fast-Trie. Here is a SkipTrie with only 2 levels of SkipList (everything else being stored in the X-Fast-Trie):

So, even in theoretically perfect conditions, the performance won’t be better than that of just X-Fast-Trie.

4. It is clear at this point the X-Fast-Trie (or the “top” structure) must be improved to see any gains in the SkipTrie. Specifically, its cache performance must be improved, as the instruction execution is insignificant. Here’s the X-Fast-Trie versus dumb queries on random memory locations (same number of mem. locations accesses as the number of X-Fast-Trie’s hash table queries):

5. Additionally, to ensure that the top of a “pure” SkipList doesn’t have hot-paths that might be constantly cached etc. (which would imply that the “top” of the SkipList is faster then the bottom), the “top” X-Fast-Trie from the SkipTrie and the top level (top 15 – the stuff that gets cut off in the SkipTrie) queries of the SkipList were compared:

Again, the time performance of the X-Fast-Trie is about 20% better than that of SkipList. This time, however, it’s not due to L3 accesses.

This leads to the conclusion, that the current X-Fast-Trie is not a sufficient improvement and some other structure must be used.