Tries
a comparison

Tries compared to Binary search trees

+ Predictable lookup time
  Worst case: O(*#letters*) vs O (*#letters* log *#entires*) for BST
+ More space efficient for short keys
+ No balancing needed

- Lose their advantages for non optimal keys (floats)
- Elaborate indexing needed for large key spaces (Unicode)
- Everyone knows BST

Tries compared to Hash tables

+ Predictable lookup time

  Worst case: O(*#letters*) vs (*#entries*) for HT

+ Support ordered iteration

+ Easy lookup of all entry with a given prefix

+ No rebuild needed (lower worst case latency)

+ No hash function needed

- Lose their advantages for non optimal keys (floats)

- Elaborate indexing needed for large key spaces (Unicode)

- Less space efficient

- Everyone knows hash tables

Conclusion

- Tries are useful for relatively-dense list of words like English dictionary or spell checker.

Footnote:

Some of the drawbacks can be alleviated by more optimized Trie data structures like Radix tree.

Sources

http://en.wikipedia.org/wiki/Trie
http://en.wikipedia.org/wiki/Talk:Trie
http://stackoverflow.com/questions/245878/how-do-i-choose-between-a-hash-table-and-a-trie-prefix-tree