Abstract

In the current scenario, with the transpiring big data explosion, data sets are often too large to fit completely inside the computers’ internal memory. In efficient processes, speed is not an option, it is a must. Hence every alternative is explored to further enhance performance, by expanding in-place memory storage that enables more data to be resident in the memory, eliminating operation latency, and even deploying an in-memory database (IMDB) system where all the data can be kept in memory. However, the technique of in-memory data handling is still at an infant stage and not viable in the current scenario. To tackle this problem a hierarchical hashing scheme is discussed where only one component of a big data structure resides in the memory. In this paper two data structures are explored: 1) Map which is implemented as self-balancing binary search trees or more commonly Red Black Trees and 2) Unordered Map which is based on hashing with chaining technique. Serialization and deserialization operations are also performed to free the internal memory and preserve the data structure object for later use. Operations such as read, write are performed, along with documentation of the results and illustrations of visual representations of the two algorithmic
References

1. Jeffrey Scott Vitter, Purdue University "Dealing with massive data" 


8. Standard C++ Library reference: 


Index Terms

Computer Science Programming Language

Keywords

Map, Unordered Map, Boost C++, Serialization, Hierarchical hashing, Memory management