Subject: Re: Map implementation
Posted by mirek on Wed, 10 Apr 2019 15:37:07 GMT
View Forum Message <> Reply to Message

Novo wrote on Wed, 10 April 2019 17:09cbpporter wrote on Wed, 10 April 2019 05:39I also talked with a colleague and his hashmap version is supposedly 3x+ faster than stl, so it looks like it is very possible to greatly outperform it.

IMHO, it is impossible to create one ideal hash table which will greatly outperform STL in all possible scenarios.

That is probably true, but mostly because at some point the limiting factor becomes cache / memory performance.

Quote:
Let's take a look at two scenarios.

1. One million hash tables containing one hundred records.
2. One hash table containing one hundred million records.

You will need two completely different implementations in these cases.

That might be true, however I do not see a way how to improve Index for either (I see some accumulated knowledge how to improve it for both, but that is another story).

Quote:
1. Add data once and search for data most of the time.
2. Add/remove data most of the time and search for it occasionally.

Ditto.

About the only thing that is in question is how to deal with collisions. Some advanced hashmaps might e.g. use binary trees to resolve collisions. I believe that it is not an overal gain (and the fact that it is not widely used in industry makes it likely) and that we should rather invest time to investigate proper hashing techniques.

Anyway, the real benchmark would be to create real world scenario and test there. IMO U++ Index/VectorMap wins that easily.

Mirek