Subject: Re: Map implementation
Posted by mirek on Wed, 10 Apr 2019 09:07:09 GMT
View Forum Message <> Reply to Message

Update: Upon further investigation... The proposed change is not perfect. Better solution:

```
inline dword FoldHash(dword h)
{
 return SwapEndian32(2833151717 * h);
}
```

Interestingly, in this particular case it runs slower. It took me a couple of hours to figure out why: It is cache issue. This much better FoldHash actually spreads hashes nicely through hash space (thus causing cache misses), while the previous one tended to put them close (cache hits). It had accidentally fixed this particular benchmark's collisions, but would probably fail in some other scenario.

Conclusion: At this number of elements, the benchmark is memory bound (for int), so well working hasmap will perform similary to std::set (actually, I think the benchmark favors std::set here, as sequential numbers will repeat the same path in the binary tree, which is more cache friendly).

Original FoldHash was too simplistic for ink keys, this new version should be harder to attack.

In future version of Index, I will try to randomize FoldHash (and other hashing ops).

Mirek